



ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits

- [Other Information Sources](#)
- [Books](#)
- [Backgrounds](#)
- [Pages](#)
- [Menus](#)
- [Graphics](#)
- [Text](#)
- [Objects](#)
- [Multimedia](#)
- [Printing](#)
- [File Management](#)
- [Dialog Boxes](#)
- [Windows](#)
- [Accessing Other Applications](#)
- [Other](#)

Listserv

LISTSERV stands for "list server". LISTSERV is a mailing-list server which is designed to make group communication easier. The address format is "List userid@node" to which mail and files must be sent in order to be redistributed to the list. The first part ("userid") will always be the list name (eg., TOOLB-L), while the second part ("node") is the node name of the LISTSERV server, eg., TOOLB-L@UAFSYSB.UARK.EDU

LISTSERV@UAFSYSB.UARK.EDU is the network address of the LISTSERV server. **This is the network address you must send commands to.**

Credits



ToolBook FAQ (c) Allan Christie 1993

Internet: Allan.Christie@UniSA.Edu.Au

Mail: Allan Christie
Centre for University Teaching and Learning
University of South Australia
GPO Box 2471
ADELAIDE SA 5001
AUSTRALIA

This document has been compiled from postings to the ToolBook [listserv](#). Many thanks go to large number of individuals who have been prepared to share their experiences and, yes, even their code in this forum. I hope you find this FAQ helpful but remember this is version 1 and there will be errors, omissions, inaccuracies, etc. Please assist me to improve this Help file by forwarding comments, suggestions, corrections, etc. to me.

This document is compilation copyright © 1993 by Allan Christie. It may be freely copied and/or distributed in its entirety as long as this copyright notice is not removed. It may not be sold for profit or incorporated into commercial products without the author's written permission.

If you have any comments, suggestions, corrections, etc. please e-mail to Allan.Christie@UniSA.Edu.Au



ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.

▼ Expand

Credits

- Other Information Sources
 - Asymetrix
 - Books
 - Courseware
 - Distributors
 - FTP Sites
 - Listserv
- Books
- Backgrounds
- Pages
- Menus
- Graphics
- Text
- Objects
- Multimedia
- Printing
- File Management
- Dialog Boxes
- Windows
- Accessing Other Applications
- Other

Asymetrix

The following information is taken from a text file on the Asymetrix BBS with amendments to include technical support available through the Internet.

ASYMETRIX TECHNICAL SUPPORT ONLINE

Asymetrix is committed to providing the highest quality technical support possible, and to that end maintains a number of forums on various online services in addition to the support offered via this BBS. There are no fees or restrictions on this support other than those imposed by the services themselves.

Following is a listing of the services and forums maintained by Asymetrix Technical Support. All of these services are checked daily for messages, and all issues posted will be addressed within 48 hours, but usually much more quickly. Asymetrix Technical support can be reached by phone at (206) 637-1600 and by fax at (206) 454-0672

ASYMETRIX BBS

Line 1: (206) 451-1173 1200-2400 baud/9600 baud US Robotics HST mode

Line 2: (206) 451-8290 2400-14400 baud v.32bis

Notes: There are four nodes, 2 on each line shown above. We use Wildcat! version 3.55 Professional. On the 451-1173 line there are two 9600 baud US Robotics HST Modems. The 451-8290 line is running 2 Practical Peripherals 14400FXSA v.32bis modems.

AMERICA ONLINE

Asymetrix can be found in the Industry Connection, which is a subsection of the Computing and Software area. For FREE SOFTWARE, and a FREE 30 DAY TRIAL PERIOD, call 1-800-827-6364 or 703-893-6288. America Online also operates a TDD (Telephone Device for the Deaf) machine for hearing-impaired members. That number is 1-800-759-DEAF.

Notes: The first month is free, which includes 5 hours of online time. The software is provided free of charge. This is a very exciting and innovative service, in which we are very pleased to be participating.

COMPUSERVE

Windows Third Party Developer 'A' Forum, Section 1

- Type: GO WINAPA

Multimedia Vendors Forum, Section 15

- Type: GO MULTIVEN

IBM Ultimedia Tools 'A' Forum, Section 5

- Type: GO ULTIATOOLS

Notes: You can also locate us online by typing 'FIND ASYMETRIX' at the command prompt. This is an excellent source of assistance and advice, not only from Asymetrix directly, but from other, very knowledgeable users all over the world.

BIX

Type "join asymetrix" and select the topic in which you would like to participate. The topics include ToolBook, Multimedia ToolBook, MediaBlitz!, and Writing and Using DLLs with ToolBook.

BIX is solely dedicated to professional people who use computers for art, business, employment, fun, personal development, profit, and for scientific, social, or product research, development, or support.

INTERNET

support@asymetrix.com

techsup@asymetrix.com

Unfortunately the Asymetrix BBS is not accessible through Internet. So if you need new filters, updated tbkmm.sbk file, etc. and don't have access to the BBS, send your regular mail address and serial number to SUPPORT@ASYMETRIX.COM. Sending your serial number is VERY important because legally only owners of Asymetrix products can get these files. The uuencoded files can then be delivered via email.

CompuServe is a registered trademark of CompuServe, Incorporated. America Online is a registered trademark of America Online, Inc. BIX is a registered trademark and is owned and operated by General Videotex Corp.

Books

BOOKS DEDICATED TO TOOLBOOK

The ToolBook Companion

Publisher: Microsoft Press ISBN # 0-936767-16-2

Author: Joseph R. Pierce

COMMENT: Written for 1.0 with no plans for an update. The problem is that ToolBook Companion is out of print and nearly impossible to find.

Building Applications with ToolBook: Create Your Own First-Rate Windows Software Quickly

Publisher: Brady Books ISBN # 0-13-092420-2

Author: Gina Smith and John Pallato

COMMENT: Jumps from basic to advanced applications. Some people have had problems with the demos. "It's the best 3rd party ToolBook 1.5 reference I've seen".

The ABC's of ToolBook for Windows

Publisher: Sybex

Author: Kenyon Brown

Comments: Sybex also released the translation in German about ToolBook 1.0 by Michael Tisher. The translated version is titled "Up & Running with ToolBook" and it came out in May 1992. Mentions ToolBook 1.5 at the very end. There is a possibility they will update it to 1.5.

Up & Running with ToolBook for Windows

Publisher: Sybex

Author: Michael Tischer

Comments: This book is in German. Was translated from "ABC's of ToolBook for Windows"

BOOKS WHICH DISCUSS TOOLBOOK

PC Magazine Windows Rapid Application Development

Publisher: Ziff-Davis Press

Author: David E.Y. Sarna, George J. Febish

Comments: Use MARVEL Programming (Modular Programming, Automatic Interfaces, Rethinking, Visual Development Environments, Extensibility, and Linking) to quickly build ToolBook applications.

Windows 3.1 End-User Programming

Publisher: Prentice-Hall/New Riders Publishing

Author: Michael Groh, et al.

Comments: Released in August 1992. Includes two sample apps. built in TB. Other tools discussed are Plus, VB, OV and Quick C. Other projects in the future will involve MM.

Welcome to Multimedia

Publisher: MIS Press

Author: Linda Tway

Comments: Released November 1992. Includes a disk with Asymetrix materials and discusses how to create a MMTB application.

COMMENT: I use this one for my Authoring class and the students have found the step-by-step instructions helpful and easy to follow.

Instant Multimedia

Publisher: John Wiley & Sons

Author: Kris Jamsa

Comments: A buyer's guide to multimedia products for the mass market, not the programmer market. Included is a discussion of MMTB. Released December 1992.

Multimedia Madness

Publisher: Prentice Hall/Sams

Author: Ron Wodaski

Comments: MMTB reviewed in his book on Multimedia. Application Space comes with the book. First copies will be in selected bookstores the week of December 14 and available widely in January 1993.

Windows 3: The Complete Reference

Publisher: Osborne McGraw-Hill, Dec. 1991

Author: Tom Sheldon

Comments: Includes a 16-page chapter entitled "DayBook and ToolBook."

Windows 3.1 End-User Programming

Publisher: New Riders Publishing, 1992.

Authors: Forrest Houlette, et al.

Comments: Includes a section on ToolBook programming and sample applications on disk

The Analytical Engine

Authors: Rick Decker and Stuart Hirshfield

Comments: Discusses the history of computing from Babbage, Von Neumann and others to issues of hardware development and ideal programming languages. All of the "hands-on" stuff is done in ToolBook, which receives a great deal of attention in many sections of the book. I have thumbed through the book and think that it would make a good text for a computer science course for non-majors. It presents a lot of ToolBook info. but in a historical and problem solving context that may not make it an ideal reference. Still it is kind of fun and comes with a diskette!

Courseware

The following 5 ToolBook files were distributed with ToolBook 1.5x and provide a wealth of online information:

QUIKTOUR.TBK

This book provides an interactive tutorial and acquaints you with the capabilities of ToolBook. The Quick Tour contains something for everyone who uses ToolBook, both readers and authors of books created with ToolBook.

OWORKBK.TBK

OpenScript Workbook covering basic concepts, elements of openScript, openScript commands, advanced topics, and openScript applications.

HELP.TBK

This book contains hundreds of topics and glossary entries.

DDETUTOR.TBK

This book describes and demonstrates using ToolBook with DDE. The Excel DDE demonstration requires that excel.exe to be in your path, that excel.exe isn't already running, and that the following files are in your current working directory:

- ddedata.xls
- ddedemo.xlw
- ddechart.xlc
- ddemac.xlm

The ToolBook DDE demonstration requires that ToolBook be in your path. Also, no other instances of ToolBook can be already running.

SPEED.TBK

This book exists to help you learn how to get the best performance from your ToolBook applications. By understanding how ToolBook works and by following some simple rules, you can design books that turn pages, animate objects, and work with text more quickly.

Distributors

AUSTRALIA

Solutions Ltd.
Suite 25, Ashmore Commercial Center
207 Currumburra Road
Ashmore City QLD 4214
(61) 75 395 422
(61) 75 393 482 - fax

Infotainment Asia Pacific Pty. Ltd.
5 Charnwood Crescent
St Kilda VIC 3182
(61) 3 5255266
(61) 3 5255482 - fax

FRANCE

Asymetrix
CNIT
CLUB SAFARI AFFAIRE/INFOMART
4E, ET (4th Floor)
2, Place De La Defense
B.P. 240,92053 Paris La Defense,
France

Christian Mercer-Laurent
(33) 1-46-92-24-53
FAX: (33) 1-46-92-24-4

FTP Sites

The following anonymous FTP sites provide a wide range of ToolBook apps., utilities, etc. many of which allow access to their scripts.

Host: asymetrix.com (hubble.com) (192.147.176.2) -- lots of ToolBook stuff - what would you expect!!

Location: /pub

Host: ftp.cica.indiana.edu (129.79.20.84) -- make sure you check for mirror sites in your own area before going to CICA.

Location: /pub/pc/win3/toolbook

Host: gandalf.iat.unc.edu (192.154.79.4) -- this site only has a small number of specific ToolBook apps. but does have a number of technical reports (multimedia, internet resources, etc.) of general interest.

Location: /pub/toolbook

Host: m-media.muohio.edu -- this is a AuthorWare Professional / Director repository of multimedia apps.! Handy to see what is being done with another development tool. Get the runtime AWP files (runawp.zip) from the utilities directory.

Location: /MultiMedia

Host: amalgame.medent.umontreal.ca (132.204.154.20) -- mainly Dentistry demos but interesting for the use of ToolBook in this area.

Location: /pub

ToolBook Listserv

Toolbook-conversation list

To subscribe; send e-mail message with one line in mail body:

```
SUBSCRIBE TOOLB-L yourfirstname yourlastname
```

to LISTSERV@UAFSYSB.UARK.EDU

For those of you having difficulties using listserv commands, I recommend getting two manuals from LISYSERV: send some mail to LISTSERV@UAFSYSB (Bitnet) or LISTSERV@UAFSYSB.UARK.EDU (Internet) and in the mail body put these two listserv commands:

```
GET LISYSERV MEMO
GET LISTDB MEMO
```

These will be of great help in providing assistance to reviewing what has been discussed in the past on TOOLB-L.

Examples of Database Commands (send to LISYSERV **not** TOOLB-L)

```
//      JOB  Echo=No
      Database Search DD=Rules
//Rules DD  *
search * in toolb-l where subject contains (dde and paradox)
print all
/*
```

```
//      JOB  Echo=No
      Database Search DD=Rules
//Rules DD  *
search selectedtextstate in toolb-l since january
index
/*
```

```
//      JOB  Echo=No
      Database Search DD=Rules
//Rules DD  *
search * in toolb-l where subject contains (dde or dll) and sender contains knowles
print all
```


ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources



Books

Automatic Running Of Book When Idle

Compressing Book

Corrupted Books

Exit - Avoiding "Save Changes" Dialog Box

Full Screen

Merging Books

Password Protection

Removing Asymetrix Banner

Returning Book To Its Default State After User Changes

ToolBook Instances



Backgrounds



Pages



Menus



Graphics



Text



Objects



Multimedia



Printing



File Management



Dialog Boxes



Windows



Accessing Other Applications



Other

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources



Books



Backgrounds

Finding Out A Book's Backgrounds

Most Space Efficient Method To Store Bitmaps



Pages



Menus



Graphics



Text



Objects



Multimedia



Printing



File Management



Dialog Boxes



Windows



Accessing Other Applications



Other

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources



Books



Backgrounds



Pages

Inserting A New Page 1 In A Book

Navigating To A Help Page

LeavePage Error Message

- [Restore Page](#)
- [Verifying A Page's Existence](#)
- [Menus](#)
- [Graphics](#)
- [Text](#)
- [Objects](#)
- [Multimedia](#)
- [Printing](#)
- [File Management](#)
- [Dialog Boxes](#)
- [Windows](#)
- [Accessing Other Applications](#)
- [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.

[Credits](#)

- [Other Information Sources](#)
- [Books](#)
- [Backgrounds](#)
- [Pages](#)
- [Menus](#)
- [Activate / Deactivate MenuItem](#)
- [Adding Accelerator Key For MenuItem](#)
- [Disabling Function Keys](#)
- [Using The PopMenu Function](#)
- [Graphics](#)
- [Text](#)
- [Objects](#)
- [Multimedia](#)
- [Printing](#)
- [File Management](#)
- [Dialog Boxes](#)
- [Windows](#)
- [Accessing Other Applications](#)
- [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources



Books



Backgrounds



Pages



Menus



Graphics

Bitmap Rescaling In Different Resolutions

Bitmap Window Scrolling

Importing Graphics Into ToolBook

Importing Icons Into Toolbook

Importing Macintosh Graphics Into ToolBook

Problems Displaying Bitmaps - Palette Shifting

Removing Line Border Around Imported Graphic

tbkBitmap()



Text



Objects



Multimedia



Printing



File Management



Dialog Boxes



Windows



Accessing Other Applications



Other

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



[Other Information Sources](#)

[Books](#)

[Backgrounds](#)

[Pages](#)

[Menus](#)

[Graphics](#)

[Text](#)

[Determining If Text Will Fit In One Line In Text Field](#)

[Determining Which Word Is Selected](#)

[Different Text Colors In Same Field](#)

[Equations](#)

[Hotwords - Modifying Properties](#)

[Hotwords - Relationship With Text](#)

[Importing Foreign Language Text](#)

[Maintaining Text Format When Copying Between Text Fields](#)

[Position Of Blinking Cursor In A Text Field](#)

[Placing Carriage Return In Text Field Using Script](#)

[Scrolling Multiple Fields](#)

[Selecting Text Lines In A Field](#)

[Sort Text In Field](#)

[Superscript/Subscript Fonts](#)

[Unchangeable But Selectable Field](#)



[Objects](#)

[Multimedia](#)

[Printing](#)

[File Management](#)

[Dialog Boxes](#)

[Windows](#)

[Accessing Other Applications](#)

[Other](#)

ToolBook FAQ (Frequently Asked Questions)















To Learn how to use Help, press F1.



[Credits](#)



[Other Information Sources](#)







-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
- [Avoiding Repetitive Tasks](#)
- [Deactivating Button](#)
- [Flashing Object](#)
- [Global Changes To Scripts](#)
- [Layering Order](#)
- [Modifying Group Components](#)
-  [Multimedia](#)
-  [Printing](#)
-  [File Management](#)
-  [Dialog Boxes](#)
-  [Windows](#)
-  [Accessing Other Applications](#)
-  [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.

[Credits](#)

-  [Other Information Sources](#)
-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
-  [Multimedia](#)
- [AVI PlayBack Window](#)
- [Executing Multimedia Event Without Locking Screen](#)
- [Moving and Positioning An Animation Window](#)
















- [Playing Sound Through Non-MCI Device](#)
- [Playing Sound](#)
- [Stopping An Animation](#)
- [Multimedia ToolBook](#)
-  [Printing](#)
-  [File Management](#)
-  [Dialog Boxes](#)
-  [Windows](#)
-  [Accessing Other Applications](#)
-  [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



[Credits](#)

















-  [Other Information Sources](#)
-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
-  [Multimedia](#)
-  [Printing](#)
- [Printing Bitmaps](#)
- [Printing Text In Fields In Multiple Pages](#)
- [Printing Text Of Ordinary Field](#)
-  [File Management](#)
-  [Dialog Boxes](#)
-  [Windows](#)
-  [Accessing Other Applications](#)
-  [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits

-  [Other Information Sources](#)
-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
-  [Multimedia](#)
-  [Printing](#)
-  [File Management](#)
-  [Changing Directory](#)
-  [Dialog Boxes](#)
-  [Windows](#)
-  [Accessing Other Applications](#)
-  [Other](#)


















ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits
















-  [Other Information Sources](#)
-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
-  [Multimedia](#)
-  [Printing](#)
-  [File Management](#)

-  [Dialog Boxes](#)
 - [ASK And REQUEST Dialog Boxes](#)
 - [Dialog Boxes - General Overview](#)
 - [Keep Dialog Box Visible And Modify ListBox Contents](#)
 - [Setting Properties Of Dialog Buttons](#)
 - [Sorting Text In ListBox](#)
 - [Using Icons In Dialog Boxes](#)
-  [Windows](#)
-  [Accessing Other Applications](#)
-  [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.

[Credits](#)

-  [Other Information Sources](#)
-  [Books](#)
-  [Backgrounds](#)
-  [Pages](#)
-  [Menus](#)
-  [Graphics](#)
-  [Text](#)
-  [Objects](#)
-  [Multimedia](#)
-  [Printing](#)
-  [File Management](#)
-  [Dialog Boxes](#)
-  [Windows](#)
 - [Always On Top](#)
 - [Enable / Disable Window](#)
 - [Exiting Windows From Toolbook](#)
 - [Prevent Switching Windows Using Alt-Tab](#)
 - [Removing Windows Styles](#)
-  [Accessing Other Applications](#)
-  [Other](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources



Books



Backgrounds



Pages



Menus



Graphics



Text



Objects



Multimedia



Printing



File Management



Dialog Boxes



Windows



Accessing Other Applications

ToolBook To ToolBook (not DDE)

ToolBook To PaintBrush (not DDE)

ToolBook To Calculator

ToolBook To Excel

ToolBook To Word For Windows

ToolBook To Access

Avoid Launching Active Applications

Sizing Other Applications From ToolBook

Determining If Other Applications Are Running



Other

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



[Other Information Sources](#)

[Books](#)

[Backgrounds](#)

[Pages](#)

[Menus](#)

[Graphics](#)

[Text](#)

[Objects](#)

[Multimedia](#)

[Printing](#)

[File Management](#)

[Dialog Boxes](#)

[Windows](#)

[Accessing Other Applications](#)

[Other](#)

[Changing The Field Delimiter In An ASCII File](#)

[Conversion Of HyperCard Stacks To ToolBook Books](#)

[Debug Window](#)

[Determining Runtime Or Full Version ToolBook](#)

[Developers' Utilities](#)

[Gang Screen](#)

[Limitations](#)

[Manipulating Scripts In Runtime ToolBook](#)

[Network](#)

[Runtime Distribution](#)

[Runtime ToolBook Errors](#)

[Script Tips](#)

[Setting Timers](#)

[ToolBook Demo - TestDrive](#)

[ToolBook Versions](#)

ToolBook FAQ (Frequently Asked Questions)

To Learn how to use Help, press F1.



Credits



Other Information Sources

Asymetrix

Books

Courseware

Distributors

FTP Sites

Listserv



Books

Automatic Running Of Book When Idle

Compressing Book

Corrupted Books

Exit - Avoiding "Save Changes" Dialog Box

Full Screen

Merging Books

Password Protection

Removing Asymetrix Banner

Returning Book To Default State After User Changes

ToolBook Instances



Backgrounds

Finding Out A Book's Backgrounds

Most Space Efficient Method To Store Bitmaps



Pages

Inserting A New Page 1 In A Book

Navigating To A Help Page

LeavePage Error Message

Restore Page

Verifying A Page's Existence



Menus

Activate / Deactivate MenuItem

Adding Accelerator Key For MenuItem

Disabling Function Keys

Using the PopMenu Function



Graphics

Bitmap Rescaling In Different Resolutions

Bitmap Window Scrolling
Importing Graphics Into ToolBook
Importing Icons Into Toolbook
Importing Macintosh Graphics Into ToolBook
Problems Displaying Bitmaps - Palette Shifting
Removing Line Border Around Imported Graphic
tbkBitmap()



Text
Determining If Text Will Fit In One Line In Text Field
Determining Which Word Is Selected
Different Text Colors In Same Field
Equations
Hotwords - Modifying Properties
Hotwords - Relationship With Text
Importing Foreign Language Text
Maintaining Text Format When Copying Between Text Fields
Position Of Blinking Cursor In A Text Field
Placing Carriage Return In Text Field Using Script
Scrolling Multiple Fields
Selecting Text Lines In A Field
Sort Text In Field
Superscript/Subscript Fonts
Unchangeable But Selectable Field










Objects
Avoiding Repetitive Tasks
Deactivating Button
Flashing Object
Global Changes To Scripts
Layering Order
Modifying Group Components



Multimedia
AVI PlayBack Window
Executing Multimedia Event Without Locking Screen
Moving and Positioning An Animation Window
Playing Sound Through Non-MCI Device
Playing Sound
Stopping An Animation



Multimedia ToolBook
Printing
Printing Bitmaps
Printing Text In Fields In Multiple Pages

-  [Printing Text Of Ordinary Field](#)
-  [File Management](#)
-  [Changing Directory](#)
-  [Dialog Boxes](#)
- [ASK And REQUEST Dialog Boxes](#)
- [Dialog Boxes - General Overview](#)
- [Keep Dialog Box Visible And Modify ListBox Contents](#)
- [Newdialog.tbk Features](#)
- [Setting Initial Focus](#)
- [Setting Properties Of Dialog Buttons](#)
- [Sorting Text In ListBox](#)
- [Using Icons In Dialog Boxes](#)
-  [Windows](#)
- [Always On Top](#)
- [Enable / Disable Window](#)
- [Exiting Windows From Toolbook](#)
- [Prevent Switching Windows Using Alt-Tab](#)
- [Removing Windows Styles](#)
-  [Accessing Other Applications](#)
- [ToolBook To ToolBook \(not DDE\)](#)
- [ToolBook To PaintBrush \(not DDE\)](#)
- [ToolBook To Calculator](#)
- [ToolBook To Excel](#)
- [ToolBook To Word For Windows](#)
- [ToolBook To Access](#)
- [Avoid Launching Active Applications](#)
- [Sizing Other Applications From ToolBook](#)
- [Determining If Other Applications Are Running](#)
-  [Other](#)
- [Changing The Field Delimiter In An ASCII File](#)
- [Conversion Of HyperCard Stacks To ToolBook Books](#)
- [Debug Window](#)
- [Determining Runtime Or Full Version ToolBook](#)
- [Developers' Utilities](#)
- [Gang Screen](#)
- [Limitations](#)
- [Manipulating Scripts In Runtime ToolBook](#)
- [Network](#)
- [Runtime Distribution](#)
- [Runtime ToolBook Errors](#)
- [Script Tips](#)

Setting Timers

ToolBook Demo - TestDrive

ToolBook Versions

Automatic Running of Book When Idle

Use the idle message. The example below assumes that if idle messages are more than 1/2 second apart, then the user must be interacting with the application.

```
to handle enterBook
  system s_lastTime,s_startTime
  linkDLL "user"
    DWORD GetTickCount()
  end
  set s_lastTime to GetTickCount()
  set s_startTime to 0
  forward
end

to handle idle
  system s_lastTime,s_startTime, s_mode
  local currentTime
  set currentTime to GetTickCount()
  if currentTime - s_lastTime > 500 -- the users has been interacting
    set s_lastTime to currentTime
  else
    if currentTime - s_startTime > 300000 -- 5 minutes of
      -- inactivity
      set s_mode to "slideShow"
      send startSlideShow
    end
  end
end

end
```

Then you need to have a keyDown handler and/or a buttonUp handler at the book level that will stop the slide show and go back to interactive mode (don't forget to forward both of these so that they reach this level)

```
to handle keyDown -- do the same thing for buttonUp
  system s_mode
  if s_mode = "slideShow"
    set s_mode to "interactive"
    send startInteractive
  else
    forward
  end
end
```

end

Exit - Avoiding "Save Changes" Dialog Box

I am running a read-only ToolBook application. Is there any way of suppressing the "Save Changes" dialogue when the application terminates?

1. set sysChangesDB to FALSE
This translates as set the changes dialog box to not shown when you exit. This has the effect of allowing the user to exit without being prompted to save changes.
2. set sysSuspendMessage to TRUE -- (before the saveAs)

Compressing Book

1. When you want to compress a ToolBook application, you should select the "Save As" option from the File menu and save your application under a DIFFERENT name. This will clear the application of any unnecessary bytes. Then of course delete or rename your original, and renaming your new one to the original name would help.
2. The easiest way to do this is to copy a script called 'to handle compactBook' from the TB manual (can't remember which one or where). It saves the original book as x.tbk and then again saves this x.tbk as the-name-of-the-original.tbk. If you have it in your book script or system book, you can start it from the command window with 'send compactBook'.

Corrupted Books

When attempting to access a page a message popped up: eg., 'General Protection Fault in module TBKCOMP.DLL at 0003:0858'

Two solutions in order of preference:

1. Before you go deleting the whole page (which may lose a lot of work), you may just want to look at or delete the enterPage handler if there is one. Perhaps the thing causing the error is there. From any page you can type "Edit script of page <name>" in the command window.
2. Copy remaining pages into a new book, one at a time. The old book is then discarded.

I have got a message like "too many objects" (or something) when I try to save my book. It doesn't help anymore to save it, I can't get rid of the message even if I delete all objects of a book and save it again!

1. It sounds like you may have a little corruption. I'd try IMPORTING the book into a new blank book. That usually fixes it.

A RELATED TOPIC:

One of my students has the following problem. He created a page with a bitmap on it and now is unable access the page because it requires too much memory. How can this page be deleted?

There is a file on the Asymetrix BBS and FTP site called XFERPGS.ZIP. This book will allow you to copy a book to a new book without specified pages.

Another possibility is to change your video depth and resolution. ToolBook has to realize that bitmap in memory--at the resolution of the screen, so a bitmap that would easily fit into memory at 640x480x4bit (standard VGA) may not fit at 1280x1024x24bit. If you can shift to MCGA or Hercules, it will take a lot less memory, and you can probably get to it.

Full Screen

Does ToolBook provide a way to display a page in a full screen window?

There are two ways of doing this:

1. link with the user.dll and find the display size:

```
linkDLL "user"  
int GetSystemMetrics(int)  
end linkDLL
```

-- Get width and height of window

```
set width to GetSystemMetrics (0)  
set height to GetSystemMetrics (1)
```

-- Get height of caption bar

```
set capHeight to getSystemMetrics(4)
```

-- Get width and height of resize borders

```
set bwidth to GetSystemMetrics (32)  
set bheight to GetSystemMetrics (33)
```

-- Open the window to maximum width and height

```
set bounds of mainWindow to \  
-1*bwidth,-1*bheight-capHeight,width+bwidth,height+bheight  
unlinkDLL "user"
```

2. the other is to just assume a standard VGA display and size your pages slightly larger so as to "Hide" the caption bar and frame. You must still hide the menubar and scrollbar yourself, and clearly inform all users to use VGA mode, even on SVGA monitors, eg.,

```
send sizeToPage  
set bounds of mainWindow to -4,-24,644,484
```

Apparently Asymetrix.com has a book on their ftp site called Fullscrn.tbk which may provide extra information.

Merging Books

There is a book on the Asymetrix BBS entitled merge.tbk to assist with this task.

Removing Asymetrix Banner

I would like to hide the small Asymetrix banner or totally remove it from appearing when ever I execute a book.

To remove the banner is very simple and "not too expensive". Asymetrix provides what they call the "logoless runtime" to participants in the Developer Services Program. You pay \$600/year and get special support, discounts on their consulting services, etc., and get the logoless runtime. You must also sign an agreement stating that you will place the Asymetrix Logo and copyright notice in the documentation and packaging.

ToolBook Instances

There are two solutions to determining if another instance of a ToolBook is already running. The second is certainly more elegant and probably the preferred solution.

Simple script using getRemote statement to determine if an application is running.

```
to get isBookRunning bookName
  clear sysError
  getRemote "this book" application toolbook topic bookName
  if "OK" is in sysError
    return TRUE
  else
    return FALSE
  end
end
end
```

Here is a script to close a book if another instance of it is already running. It uses tbkBMP to manipulate a true global variable as a flag that says, "I'm running", when your book starts up, and clearing this flag when it is closed. The windows function in user.exe is used to bring an already open book to the top of the windows desktop. It could be modified so that many books could use it, each one adding its own path and filename to variable "g_open". In this example TRUE is used for its value.

```
-- You must have multimedia toolbook and tbkmm.sbk in startup sysbooks
to handle enterBook
  forward
-- g_open is your flag; it can be set to true/false, or to a name of a
specific book
-- test the flag: was it set by an earlier start-up?
  get getGlobalVar(g_open)
  if it is NULL
-- set the flag to TRUE
    get setGlobalVar(g_open,TRUE)
  else
-- the flag has already been set
    request "This book is already open."
    send exit
    linkDLL "user"
      int bringWindowToTop(word)
    end linkDLL
-- this will bring the first ToolBook app in the list to front:
    getRemote "sysWindowHandle" application toolbook
```

```
        get bringWindowToTop(it)
-- if you want a specific ToolBook app, add 'topic' to getRemote
    end if
end enterBook

to handle leaveBook
    get setGlobalVar(g_open, NULL)
    forward
end leaveBook
```

Inserting a New Page 1 in a Book

How does one easily insert a new Page 1 in a book?

There are a number of methods available to do this:

1. Insert the new page 1 after the current page 1 & then cut the current page 1 and paste it after the new page 1
2. An easier way would be to set the page number in the Page Properties window.
3. Another possibility, would be to do it on the fly with the following command:

```
set the pageNumber of this page to 1
```

Navigating to a Help Page

There are a number of ways of using Help pages in ToolBook

1. Have an actual Help page/s in you ToolBook application and use the "back" command to return to the place they were directly before clicking the button that has the back command. This should work well for you as long as your help is not more than one page long. However, be sure you do not have sysHistory turned off, which may be appropriate to conserve memory in other applications.

```
to handle buttonup
    send back
end
```

Use the overhead menu or a button for navigating to and from the Help page/s.

2. Create a separate book (help.tbk) that a user would call just as help works in Toolbook or Windows. One advantage of a separate book is that you can size the page differently (smaller) to appear above the reader's present page. However, to do this, you must also place the following in the book script:

```
to handle executeremote
    executeremote "send exit" topic "help.tbk"
end
```

```
to handle activateinstance
    executeremote "send exit" topic "help.tbk"
end
```

Without the "executeremote" command, if a reader clicked outside the boundaries of the Help.tbk page (i.e., assuming the Help.tbk page is smaller than the main toolbook instance), the Help.tbk page would disappear but would remain active (hidden behind the main toolbook instance and the Window screens). When the reader called up Help again, another instance of Help would be activated. If your reader is working on a network, multiple calls on the Help.tbk would quickly cause a disaster. The executeremote commands protects against this.

3. Using the Windows Help System with ToolBook

```
to handle buttonUp
    linkDLL "USER"
        INT WinHelpIndex = WinHelp(WORD, STRING, WORD, DWORD)
        INT WinHelpKey = WinHelp(WORD, STRING, WORD, STRING)
    end linkDLL
```

```

-- set up the filename
    set helpfile to "C:\TOOLBOOK\OPNSCRPT.HLP"
-- Prompt for a keyword: Of course, you could specify the keyword
    ask "Keyword?"
    if sysError is "CANCEL"
        break
    end
    if it is not NULL
        get winHelpKey(sysWindowHandle, helpfile,261, it)
            -- 257 = HELP PARTIALKEY
    else
        get winHelpIndex(sysWindowHandle, helpfile,3, 0)
            -- 3 = HELP INDEX
    end
end

-- According to the SDK documentation, applications that use
-- help should call this function before closing so that their
-- help window can be closed if open.

to handle exit
    linkDLL "USER"
        INT WinHelpQuit = WinHelp(WORD, STRING, WORD, DWORD)
    end linkDLL
    get winHelpQuit(sysWindowHandle, "",2,0)
        -- 2 = HELP QUIT
    forward
end

```


LeavePage Error Message

A field has mouseenter and mouseleave handlers to show and hide another field. If the mouse is left positioned on that field, and I go to another page I get an error on the mouseleave handler that it cannot find the field to be hidden since I am now on a different page. How can I avoid this?

The way to handle this situation is to make the hide handler very specific...

```
hide field "message" of page Introduction
```

OR set sysSuspend to false

Restore Page

If page is static, you can do this once:

```
to handle storestate
  set vobjs to objects of this page
  while vobjs is not null
    pop vobjs into vobj
    set pBounds of vobj to bounds of vobj -- store bounds into user
property
  end
end
```

Then restore it (typically in a leave page)

```
to handle leavepage
  set syslockscreen to TRUE -- don't let the reader see this mess
  send restorestate
end
```

```
to handle restorestate
  set vobjs to objects of this page
  while vobjs is not null
    pop vobjs into vobj
    set vbounds to pBounds of vobj
    if vBounds is not null
      set bounds of vobj to vBounds
    end
  end
end
end
```

Disabling Function Keys and Accelerator Keys

How can I easily disable the menu Function keys?

If you remove the individual menus many of the Function keys and Accelerator keys will be disabled:

```
to handle enterBook
  remove menu "file" at reader
  remove menu "edit" at reader
  remove menu "text" at reader
  remove menu "page" at reader
  remove menu "help" at reader
end enterBook
```

You must use "remove menu ..." rather than "hide menuBar" because "hide menuBar" leaves the accelerator table intact. It is probably better to remove all menus and add them back if you need menu functionality.

To prevent a user exiting the program by using CTRL-X simply insert a "to handle exit" handler in your book script.

How can I prevent users accessing the status box by pressing F12?

```
to handle enterBook
  hide statusBox
  ...
end

to handle toggleStatus -- invoked when F12 is pressed
  -- Don't show the page numbers
  break
end
```

How do I ask the user to press F1 and not have ToolBook go to its Help?

F1 sends the doHelp message, so to trap it, simply handle dohelp:

```
to handle doHelp
  request "and you thought you wanted help!"
end
```

On the Asymetrix BBS, there is a file named FKEYS.ZIP that shows how to trap all of the function keys except the F10 key. The F10 key is much harder than the rest, but there is a separate file on the Asymetrix BBS called F10KEY.ZIP that shows a method of trapping this key, but the method requires removing the system menu and hiding the menu bar. To trap the ALT key see TRAPALT.ZIP on the Asymetrix BBS.

In a development environment I can't toggle back to author level if I remove all the function keys? Do I need to write a handler for it in the book script?

The edit menu has been removed so Toolbook doesn't respond to the F3 key. You could put in your own handler to respond to the F3 key (see below) or, leave the Edit menu in and remove all its menu items except for "Author". This way F3 key still works.

```
to handle keydown fKey
  if fKey is KEYF3
    send Author
  else
    forward
  end
end
```

One of the enhancements I would like to add to the application is the use of Function Keys, however, when I try to intercept them they still start the Toolbook functions associated with them.

At book level, put in a handler for the message that function key sends. Don't forget - DON'T forward the message if you don't want the default action to occur.

For example, if you want to use F3, write a handler for the reader or author messages:

```
to handle reader
  send author
end reader
```

Using the PopMenu Function

Along with pull-down menus, I also have need for some submenus for several subjects. What sort of modification would that require in scripting?

Using PopMenu() from TBKWIN.DLL

Here's a script that has 3 menuitems each with submenus of varying sizes. The key ingredient is the squigly brackets "{ }". The brackets tell you where each submenu begins and ends. Notice that you put a left bracket before the submenu parent, and a right bracket in the next item after the last submenuitem.

Them items in the menulist are numbered sequentially. Put the following in your page script:

```
to handle buttonDown loc
  linkDLL "TBKWIN.DLL"
  INT PopMenu (WORD, STRING, INT, STRING, STRING, STRING)
  end
  set menuList to "{A,A1,A2,},{B,B1,B2,B3,B4,},{C,C1,C2,C3,}"
  get
  PopMenu (sysWindowHandle, sysPageScroll, sysMagnification, loc, menuList, "")
  if it > 0
    conditions
      when it = 2
        request "You chose A1"
      when it = 3
        request "You chose A2"
      when it = 6
        request "You chose B1"
      when it = 7
        request "You chose B2"
      when it = 8
        request "You chose B3"
      when it = 9
        request "You chose B4"
      when it = 12
        request "You chose C1"
      when it = 13
        request "You chose C2"
      when it = 14
        request "You chose C3"
    end
  end
```

end
end

Adding Accelerator Key for MenuItem

To add an accelerator key to the menu do is something like this:

```
put MyMenu & tab & "Ctrl+D" into It
add menuItem It to menu file
```

and you need to write a handler for the key sequence too:

```
to handle keydown, key, isshift, isctrl
  if key = keyD and isctrl = true
    send MyMenu
  else
    forward
  end
end
```

Importing Graphics into ToolBook

As supplied, ToolBook 1.5x can only import graphics files of five different types: .BMP, .CGM, .EPS, .DRW, .TIF

If you have installed a Microsoft package such as Word for Windows or Excel then you will have the "msapps" subdirectory in your "windows" directory. This includes extra graphics filters for importing many types of files (including .PCX, .PIC, and .WMF).

To make ToolBook use these filters you need to edit your WIN.INI file. Just copy and paste lines from the [MS Graphic Import Filters] section into the [ToolBook Filters] section. Each line defines one filter type. Take care not to duplicate the ones that already exist.

There is a ToolBook imposed maximum of 12 filter types.

Next time you run ToolBook, you'll get the extra options in the File|Import Graphic option under the Filters combo box.

I am having trouble importing a .EPS file?

There are different .EPS file formats. For less complicated .EPS files, I understand that the filter from MS Word for Windows will work with Toolbook. You just have to change the filter reference in the WIN.INI file as described above.

```
[ToolBook Filters]
```

```
Encapsulated Postscript(.EPS)=c:\windows\msapps\grphflt\epsimp.flt,EPS
```

Applications such as Word for Windows, Excel, Corel Draw, etc. tend to come with EPS filter files. The one in the above example came from Word for Windows which created the MSAPPS\GRPHFLT subdirectory.

Bitmap Rescaling in Different Resolutions

Bitmaps (eg., .bmps) are not a "scaleable" image. If you were to use something like a .wmf or .cgm image you could scale it like you want.

You can't resize .bmps in toolBook. The reason that you can resize after you have pasted into paintbrush is that it puts a metafile on the clipboard, which toolbook can resize.

If you use bitmaps, then develop in the highest resolution possible. This means that moving to VGA will provoke Toolbook to scale the bitmaps down, so that they fit the same size of object... but it's not as much of a mess as TBK's attempt to scale VGA bitmaps up to SVGA!!

Bitmap Window Scrolling

This is not supported by TBKBMP.DLL. But here is a brute force method for doing it. This is from the Asymetrix BBS (BMPSCL2.ZIP).

This example requires a second bitmap (BACK.BMP) to use as the parent window which actually has the scrollbars on it.

```
to handle buttonUp
  system x,y,height,width,rectHeight,rectWidth,s_scrollHandle, \
    s_clipHandle,clientWidth,clientHeight,myRightSide, \
    myTopSide,MyBottomSide,xScrollRange,yScrollRange

linkDLL "tbkwin.dll"
  STRING clientFromPage (STRING,INT,STRING)
end linkDLL

linkDLL "tbkfile.dll"
  INT fileExists (STRING)
end linkDLL

linkDLL "user"
  INT GetSystemMetrics (INT)
  LONG SetWindowLong (WORD,INT,LONG)
  INT GetClientRect (WORD,POINTER)
  INT SetScrollRange (WORD,INT,INT,INT,INT)
  INT GetScrollPos (WORD,INT)
  INT SetScrollPos (WORD,INT,INT,INT)
end linkDLL

linkDLL "kernel"
  WORD globalAlloc (WORD,DWORD)
  POINTER globalLock (WORD)
  INT globalUnlock (WORD)
  WORD globalFree (WORD)
end linkDLL

get fileExists ("back.bmp")
if it <> 1
  request "Please put the file ""back.bmp"" into the directory that" \
    && "this file, ""bmpscl2.tbk"", is in or into a directory" \
    && "that is in your path."
```

```

        break
    end if

    get fileExists ("nar.dib")
    if it <> 1
        request "Please put the file ""nar.dib"" into the directory that" \
            && "this file, ""bmpscri2.tbk"", is in or into a directory" \
            && "that is in your path."
        break
    end if

```

--Get the system metrics for the window components:
 --(All returned values are in pixels)

--The height of the window's caption bar.

```
set myTopSide to GetSystemMetrics (4)
```

--The height of the window's horizontal scroll bar.

```
set myBottomSide to GetSystemMetrics (3)
```

--The width of the window's vertical scroll bar.

```
set myRightSide to GetSystemMetrics (2)
```

--The height of the window's border.

```
set myBorderHeight to GetSystemMetrics (33)
```

--Create the window using tbkBitmap() to use as the cropping/viewing

--window, back.bmp is just a 1x1 monochrome bitmap created in paintbrush.

--This path is hardcoded right now.

```
get tbkBitmapchk("open back.bmp alias clipWnd style overlapped parent" &&
sysWindowHandle, 1, 1)
```

```
set s_clipHandle to tbkBitmap("status clipWnd window")
```

--Set the window style to be the same as an overlapped window with

--the addition of vertical and horizontal scrollbars.

```

-- WS_OVERLAPPED           0
-- WS_CAPTION              12582912
-- WS_THICKFRAME           262144
-- WS_VSCROLL              2097152
-- WS_HSCROLL              1048576
-- WS_SYSMENU              524288
-- WS_MINIMIZEBOX         131072
-- WS_MAXIMIZEBOX         65536

```

```

-- -----
-- FINAL WINDOW          16711680

get SetWindowLong (s_clipHandle, -16, 16711680)

--Open the bitmap you want to display with clipping window as parent
get tbkBitmap("open nar.dib alias scrollWnd style child parent" &&
s_clipHandle)

--Get extent of your bitmap for use in the scroll buttons
get tbkBitmap("status scrollWnd extent")
set height to item 2 of it          --Height is used in the scroll buttons to check
bounds
set width to item 1 of it          --Width is used in the scroll
buttons to check bounds

--Set size & position of Clipping window. Position is relative to the upper left
--corner of the ToolBook client window.
set wRect to clientFromPage(sysPageScroll,sysMagnification,bounds of
rectangle "clipRect")
set rectWidth to item 3 of wRect - item 1 of wRect
set rectHeight to item 4 of wRect - item 2 of wRect
get tbkBitmap("window clipWnd size" && rectWidth & "," & rectHeight)
get tbkBitmap("window clipWnd position" && items 1 to 2 of wRect)

--Get the bounds of the client area of the clipping window.
--Since items 1 and 2 of the bounds are "0", items 3 and 4
--are the width and height respectively.
set hClipWndBounds to globalAlloc(66, 20)
set lpClipWndBounds to globalLock(hClipWndBounds)

get GetClientRect (s_clipHandle, lpClipWndBounds)

set clientWidth to pointerInt (4, lpClipWndBounds)
set clientHeight to pointerInt (6, lpClipWndBounds)

--Set x & y to the initial position of the scroll window
set x to 0
set y to 0

--Set initial position of scroll window
get tbkBitmap("window scrollWnd position"&& x & "," & y)

```

--Show the scroll window first so when you show the clip window \
--you don't see it.

```
get tbkBitmap("window scrollWnd state show")  
get tbkBitmap("window clipWnd state show")
```

--Set the scroll range of both scroll bars.

```
set yScrollRange to 100  
get SetScrollRange (s_clipHandle, 1, 0, yScrollRange, 1)  
set xScrollRange to 1500  
get SetScrollRange (s_clipHandle, 0, 0, xScrollRange, 1)
```

--Translate window messages for window that is scrolled.

```
set s_scrollHandle to tbkBitmap("status scrollWnd window")  
translatewindowmessage for s_scrollHandle  
    before 2 send bmpWindowClose to self    --destroy window message  
    after 514 send bmpButtonup to self    --buttonUp message  
end
```

--Translate window messages for window that contains scroll bars.

```
translateWindowMessage for s_clipHandle  
    after 277 send vertScroll to self -- vertical scroll message  
    after 276 send horizScroll to self -- horizontal scroll message  
end
```

```
get globalUnlock (hClipWndBounds)  
get globalFree (hClipWndBounds)
```

end

--After the message WM_VSCROLL is sent to the clipping window
--this handler is processed.

```
to handle vertScroll hwnd, wmsg, wParam, lplo, lphi  
    system x,y,height,rectHeight,s_clipHandle,clientHeight, \  
        myBottomSide, myTopSide, yScrollRange  
  
    set yScrollAmount to yScrollRange * (10 / (height - clientHeight))  
    set myVertPos to GetScrollPos (s_clipHandle, 1)
```

conditions

--The vertical scroll bar up arrow has been pressed.

```
    when wParam is "0"  
        if y < 0
```

```

increment y by 10
set newVertPos to myVertPos - yScrollAmount
get tbkbitmap("window scrollWnd position" && x & "," & y)
get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
end if

```

--The vertical scroll bar down arrow has been pressed.

```

when wParam is "1"
  if y > -(height - rectHeight + myBottomSide + myTopSide)
    set newVertPos to myVertPos + yScrollAmount
    decrement y by 10
    get tbkbitmap("window scrollWnd position" && x & "," & y)
    get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
  end if

```

--Area between the thumb box and up arrow has been pressed.

```

when wParam is "2"
  if y < 0
    if y <= -(10 * 10)
      set newVertPos to myVertPos - (10 * yScrollAmount)
      increment y by (10 * 10)
      get tbkbitmap("window scrollWnd position" && x & "," & y)
      get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
    else
      set newVertPos to 0
      get tbkbitmap("window scrollWnd position" && x & "," & y)
      get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
    end if
  end if

```

--Area between the thumb box and down arrow has been pressed.

```

when wParam is "3"
  if y > -(height - rectHeight + myBottomSide + myTopSide)
    if y > (10 * 10) - (height - rectHeight + myBottomSide +
myTopSide)
      set newVertPos to myVertPos + (10 * yScrollAmount)
      decrement y by (10 * 10)
      get tbkbitmap("window scrollWnd position" && x & "," & y)
      get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
    else
      set y to -(height - rectHeight + myBottomSide +
myTopSide)
      set newVertPos to yScrollRange
      get tbkbitmap("window scrollWnd position" && x & "," & y)
      get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
    end if
  end if

```

```

        end if
    end if
--The thumb box has been dragged up or down and released.
    when wParam is "4"
        set newVertPos to lplo
        set y to (lplo / yScrollRange) * -(height - rectHeight +
myBottomSide + myTopSide)
        get tbkbitmap("window scrollWnd position" && x & "," & y)
        get SetScrollPos (s_clipHandle, 1, newVertPos, 1)
    end conditions
end

to handle horizScroll hwnd, wmsg, wParam, lplo, lphi
    system
x,y,width,rectWidth,s_clipHandle,clientWidth,myRightSide,xScrollRange

    set xScrollAmount to xScrollRange * (10 / (width - clientWidth))
    set myHorizPos to GetScrollPos (s_clipHandle, 0)

conditions
--The horizontal scroll bar left arrow has been pressed.
    when wParam is "0"
        if x < 0
            set newHorizPos to myHorizPos - xScrollAmount
            increment x by 10
            get tbkbitmap("window scrollWnd position" && x & "," & y)
            get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
        end if
--The horizontal scroll bar right arrow has been pressed.
    when wParam is "1"
        if x > -(width - rectWidth + myRightSide)
            set newHorizPos to myHorizPos + xScrollAmount
            decrement x by 10
            get tbkbitmap("window scrollWnd position" && x & "," & y)
            get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
        end if
--The area between the thumb box and the left arrow has been pressed.
    when wParam is "2"
        if x < 0
            if x < -(10 * 10)
                set newHorizPos to myHorizPos - (10 * xScrollAmount)
                increment x by (10 * 10)
            end if
        end if
    end conditions
end

```

```

        get tbkbitmap("window scrollWnd position" && x & "," & y)
        get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
    else
        set newHorizPos to 0
        set x to 0
        get tbkbitmap("window scrollWnd position" && x & "," & y)
        get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
    end if
end if

```

--The area between the thumb box and the right arrow has been pressed.

```

when wParam is "3"
    if x > -(width - rectWidth + myRightSide)
        if x > (10 * 10) -(width - rectWidth + myRightSide)
            set newHorizPos to myHorizPos + (10 * xScrollAmount)
            decrement x by (10 * 10)
            get tbkbitmap("window scrollWnd position" && x & "," & y)
            get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
        else
            set newHorizPos to xScrollRange
            set x to -(width - rectWidth + myRightSide)
            get tbkbitmap("window scrollWnd position" && x & "," & y)
            get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
        end if
    end if
end if

```

--The thumb box has been dragged left or right and released.

```

when wParam is "4"
    set newHorizPos to lplo
    set x to (lplo / xScrollRange) * -(width - rectWidth +
myRightSide)
    get tbkbitmap("window scrollWnd position" && x & "," & y)
    get SetScrollPos (s_clipHandle, 0, newHorizPos, 1)
end conditions
end

```

--dismiss the bitmap when they click it

to handle BMPbuttonUp

```
system s_scrollHandle, s_clipHandle
```

--we use a system variable for the bitmap handle here since a timer

--notification handler might send this message, and the timer notification

--handler doesn't have access to the normal window handle container

```
untranslateAllWindowMessages for s_clipHandle
```

```
untranslateAllWindowMessages for s_scrollHandle
```



```
get tbkBitmap("window clipWnd state hide")
get tbkBitmap("window scrollWnd state hide")
get tbkBitmap("close scrollWnd")
get tbkBitmap("close clipWnd")

unlinkDLL "user"
unlinkDLL "tbkwin.dll"
unlinkDLL "kernel"
end
```

Importing Icons into ToolBook

Often I want to use Windows .ico files as button icons in ToolBook. How can I easily import them into ToolBook?

If you get a shareware program called "The Cleaners" it has an application called iconclen.exe. This program allows you to view icon files one at a time (*.ico), but in addition it allows you to copy, delete and move files as you are viewing them. A special feature allows you to save any icon as a BMP file if you desire

This program is available from CICA and mirror sites:

The Cleaners (CICA: /pub/pc/win3/util/clen.zip)

Problems Displaying Graphics - Palette Shifting

Why do bitmaps flash in ToolBook?

This is called "palette shifting" and it is common to all applications that use 8-bit ("palettized") video, whether Windows, Amiga, or Mac. The way that these systems simulate photorealistic images is through a color table or palette. Essentially, your video card can show 16,777,215 colors (2^{24} , or "24-bit" color) but it can only show 256 (2^8 or "8-bit" color) at a time. So it builds a table that looks something like:

color	red	green	blue	
1	255	255	255	<-this color is white
2	255	0	255	<-this color is purple
.				
255	0	0	0	<-this color is black

and your picture, instead of using 3 bytes for each pixel (one each for red, green, and blue) uses 1 byte per pixel--the index or "color" number.

Windows has a set of routines called the palette manager, and ToolBook uses these routines to set up the hardware registers on your card with the contents of this table. Because this is a very time consuming function, and because changing the register color values causes royal blue pixels to turn to puce yellow (because the next picture uses that color), the palette manager tries to keep from changing the palette--but it doesn't succeed very often--unless the colors are all selected from a common set. And so, you get flashes in ToolBook.

In Windows, each DIB or Windows metafile can include a "palette" of 256 colors that is stored with the graphic itself. (In the Macintosh world it's called a CLUT for Color Look-Up Table). This palette is interpreted by MS Windows applications like TBK to set the selection of colors for the display device.

Now, if you place more than one palettized graphic on a single page in Toolbook, their color palettes may conflict. TBK uses the palette of the last DIB or picture created on that page. Also, if you have more than one TBK instance open at the same time with conflicting palettes, Windows resolves this conflict by using the palette information in the active window. When you are running Windows with 256 colors, ToolBook changes the color palette to match the displayed bitmap on each page. Switching between pages that have different color palette images, you "see" the switch of color palettes.

The easiest way to solve this problem is to create a custom palette (i.e. table of color information) that will contain the essential 256 colors from all of the images then apply that palette to all the images. You need to use the BITEDIT and PALEDIT programs that are a part of Video for Windows 1.0. Use the VIDEDIT utility to import all the bitmaps of an application into one video sequence and then extract a palette. BITEDIT can be used to apply the palette to each bmp or a shareware program called Paint Shop Pro 2.0 will also allow you to do this.

One procedure for creating a common palette:

- 1) Put all your bitmaps in a single directory.
- 2) Rename them so they are a sequence (bmp01.dib, bmp02.dib,..etc)
- 3) Import them into videdit as a "dib sequence."
- 4) Choose Video!Create Palette and create a palette
- 5) Copy the palette to the clipboard
- 6) Load each bitmap into bitedit and reduce its palette, using the palette from videdit.
- 7) Import the changed bitmaps into ToolBook

It should be mentioned that if your color display is capable of displaying 16-bit or 24-bit color, color palette conflicts aren't a problem. But of course, most people currently have displays (and Windows drivers) that only display 256 colors at resolutions like 640x480 or 800x600. For more details about palettized DIBs, you should read Chapter 19, "Color Palettes," in Microsoft Windows Guide to Programming.

Importing Macintosh Graphics into ToolBook

1. The import filters available on the Asymetrix BBS (we're only licensed to distribute them to registered owners of Asymetrix products so we can't put them up for anonymous FTP) will allow you to import a number of file formats including the Macintosh Quickdraw file format (PICT.)
2. *Starting with 24-bit images on the MAC and displaying in 8-bit on the PC.*
Save the MAC images on DOS formatted discs as their native format, i.e., TIF 24-bit. Then use Paint Shop Pro (ftp from ftp.cica.indiana.edu: /pub/pc/win3/desktop/pspro200.zip) to open it up as a 24-bit and then load a pre-saved palette. You can use BITEDIT and VIDEDIT (programs available in Video for Windows 1.0 package) to create a common palette in a video sequence. After using a common palette, the colors are about as good as you are going to get in 8-bit.

Removing the Line Border Around Imported Graphic

To remove the border, select the paintObject and either type "Set lineStyle of selection to 0" in the Command Window or bring up the line palette and select the "- 0 -" option.

Different Text Colors In Same Field

Can one of the words in a field be a different color?

1. There is no easy way of displaying different colors of text in same field. There are many work-arounds all of which fall down if you have a scrolling field. If not, copy the text into an overlaid field and wait for the next version of Toolbook.

2. ToolBook does not directly support changing the strokecolor of individual characters within a script. You can simulate this in a rough fashion if you are willing to use a fixed pitch font such as courier as follows:
 - (1) create a field with a fixed pitch font and enter the text
 - (2) set the strokecolor to the color you want for highlighted text
 - (3) cut and paste a copy of the field exactly on top the first
 - (4) make the new field transparent with a white fillcolor
 - (5) set the strokecolor of the new field to black
 - (6) change the text you want to highlight to spaces as the first field shows through

3. Another method is just to put a transparent colored rectangle over the text you want to color. In either case you must be concerned with synchronizing the text and color if the text is changed or scrolled.

Equations

Instead of creating equations in toolBook, consider creating your equation in a third party tool, such as Microsoft Word 2.0, and import the equation into toolBook as a bitmap or metafile. This will be a single object, and will display faster than fresh construction.

Hotwords - Modifying Properties

I have a problem. If I create hotword to text of field "XX" FROM SCRIPT (e.g. select word and send createHotword) how can I set or get any properties of that new hotword from that same script what created it?

Every time a new hotword is created, a MAKE message is sent to it. You can take advantage of that by creating a handler (eg. in the book script) that has:

```
to handle make
  if the object of target is hotword
    set lasthot of this book to target
  end
end
```

After that you can refer to the latest hotword by using the term THE LASTHOT OF THIS BOOK

OR

In a background handler:

```
to handle nameTheHotword hwName, theFieldName
  put objects of recordField theFieldName of this page into hwList
  if itemCount (hwList) = 0 then
    put "no hotwords for recordField" && theFieldName
    break nameTheHotword
  else
    step i from itemCount (hwList) to 1 by -1
      if word 1 of item i of hwList = "hotword" then
        put item i of hwList into theHW
        set the name of theHW to hwName
        break
      end if
    end step
  end if
end nameTheHotword
```

Note that this handler relies on the observation that the most recently created hotword is at the end of the list. There is no documentation to substantiate this claim. This is followed by:

```
select <some text>
send createHotword
send nameTheHotword <unique generated hotword name> <field name>
-- then do things with the hotword, including attaching a script to it
```

Make sure field is activated for Hotwords to work properly.

Hotwords - Relationship With Text

```
if the object of target is hotword
  set tloc to textfrompoint(loc)
  set locenv to characters tloc-20 to tloc+20
  --assuming hotwords are never longer than 20 chars
  --replace by something more elegant
  --this puts the text surrounding the target hotword
  --to locenv
  put offset(text of target, locenv)+tloc-20 to locstart
  -- plus or minus one maybe, these are always a drag to
  -- figure out
  put locstart+charcount(text of target) to locend
  system tarhot
  system tarloc
  set tarloc to locstart &" "& locend
  set tarhot to target
  --now you have the hotword and it's location in text
  -- in memory to be used by later scripts
end if
```

Importing Foreign Language Text

The answer to this problem is to convert a file from OEM to ANSI. This can be done in ToolBook using the functions OEMtoANSI and ANSItOEM in the windows API. They are both in keyboard.drv:

```
linkdll "keyboard"
    int AnsiToOem (string,string)
    int OemToAnsi (string,string)
end

--read record into a variable "myFrenchPhrase"
get OemToAnsi(myFrenchPhrase,myFrenchPhrase)
set text of field "display" to myFrenchPhrase
--do whatever you want....
set myFrenchPhrase to text of field "display"
get AnsiToOem(myFrenchPhrase,myFrenchPhrase)
--write the record back out
```

OR

```
linkDLL keyboard
    INT OemToAnsi (STRING, POINTER)
    INT AnsiToOem (STRING, POINTER)
end linkDLL

to get oemfromansi ansitext
    set gmem to globalAlloc(0,256*256)
    -- Sets the value of the container gmem to the file handle
    -- returned by globalAlloc when allocating global memory as
    -- 256*256 bytes of fixed memory (GMEM_FIXED)
    set pmuut to globalLock(gmem)
    -- Sets the value of the container pmuut to the long
    -- pointer to the memory object returned by locking global
    -- memory at location gmem
    --globalalloc etc. need also to be linked before this
    set i to ansiTooem(ansitext, pmuut)
    -- put the oem-version of ansitext into memory location
    -- where pointer pmuut points
    set muutettu to pointerString(0, pmuut)
    --puts the text where the pointer points into variable
    set x to globalunlock(gmem)
```

```
set x to globalfree(gmem)
  --release memory
return oemfromansi
end
```

Position Of Blinking Cursor In A Text Field

Can the position of the blinking cursor IN a text field be determined?

There is a Windows API function:

```
linkDLL "user"  
    int GetCaretPos (pointer)  
end
```

The pointer in GetCaretPos points to a point structure that you will need to allocate in Global Memory. That structure will give you the coordinates of the caret--in client coordinates. Then you can use the functions in tbkwin.dll to convert to page coords. Then you can get textfrompoint. There is a file called "caret.zip" in the directory "/pub/techsupport" on the ftp-site asymetrix.com that is an example of how to use the GetCaretPos handler.

Scrolling Multiple Fields

How can I scroll multiple fields in ToolBook?

This can be handled using the "textScrolled" handler (undocumented in the version 1 manuals) and the "scroll" property. I have a pair of fields which scroll simultaneously, only one of which, "selection", has its scroll bar showing. In that field's script I have the following handler:

```
to handle textScrolled newPos
    set scroll of field "selection" to newPos
    set scroll of field "sel_freq" to newPos
end textScrolled
```

Sort Text In Field

to handle SortText

```
system txt
set txt to text of recordfield FName
send sort 1, textlinecount (txt)
set text of recordfield FName to txt
```

end

to handle sort l,r

```
system txt
set i to l
set j to r
set x to textline ((l+r) div 2) of txt
do
  while textline i of txt < x as text; increment i; end
  while x < textline j of txt as text; decrement j; end
  if i<=j
    set y to textline i of txt
    set textline i of txt to textline j of txt
    set textline j of txt to y
    increment i
    decrement j
  end if
until i>j
if l<j; send sort l,j; end if
if i<r; send sort i,r; end if
```

end sort

Superscript/Subscript Fonts

Apparently there is are superscript and subscript fonts on the Asymetrix BBS. The following is a posting to the ToolBook listserv about this topic:

"I downloaded the super & sub scripts from Asymmetrix's BBS & they work very well in our Chem stax here at St. Kate's. I had to unzip them (so I also downloaded PKUNZIP from the same BBS). Then I just put a copy of the the font in the same directory with the Chem Books & Toolbook."

Placing Carriage Return in Text Field Using Script

How do I put a carriage return into a field using script?

CRLF is a carriage return and line feed variable. ... &CRLF could be inserted into text this way;

```
set text of field X to "Now is " &CRLF
```

Maintaining Text Format When Copying Between Text Fields

I am trying to copy a line of text from field "foo1" and put it into field "foo2". The textline in field "foo1" has two different fonts in the one line, and when it is placed in field "foo2", the whole line is pasted with just one font being used. How can I maintain the text format when copying between two fields?

Let's assume that you know that this information is in chars n to m of the field, and you just want to put it into the other field:

```
select chars n to m of text of field "source"  
send copy  
set focus to field "destination"  
send paste
```

If you want to position it elsewhere, you'll have to send a bunch of KeyDown messages to move the caret around.

OR

```
to handle pasteLine fieldRef, lineNum  
  -- fieldRef must be the uniqueName of the source field for this to work  
  set sysLockScreen to TRUE  
  set sysSuspendMessages to TRUE  
  go (words 5 to 7 of fieldRef)  
  select textLine lineNum of text of fieldRef  
  send copy  
  set focus to field "mainField2"  
  -- you can use the keyDown message to put the text at the end of the field  
  send keyDown keyEnd, FALSE, TRUE  
  send paste  
  send back  
  set sysSuspendMessages to FALSE  
end
```

OR

Copy this script into a button changing the script to represent your field names. It will copy the text slowly but with all the correct formatting.

```
to handle buttonUp  
  set syslockscreen to true  
  step I from 1 to charcount(text of field "source")
```

```
        set char I of text of field "destination" to char I of text of field
"source"
        set fontstyle of char I of text of field "destination" to fontstyle
of char I of text of field "source"
        set fontsize of char I of text of field "destination" to fontsize of
char I of text of field "source"
        set fontface of char I of text of field "destination" to fontface of
char I of text of field "source"
    end step
end buttonUp
```

Flashing Object

Use an idle handler ...

```
to handle idle
  system lastTime
  set sysTimeFormat to seconds
  if sysTime<>lastTime then
    get visible of {insert your object here}
    set visible of {object} to not it
    set sysTime to lastTime
  end
end idle
```

The above is the basics of a script that should make an object flash on and off slowly. There is a need to perform a lastTime initialisation check.

If you need something to flash faster, you'll need to use one of the Windows DLLs to get the system time in something better than seconds.

Another nice way to flash an object (say during a mouse click) is to rotate its colors by 180 degrees. Hue, as you recall, is defined from 0 to 360, with 0 and 360 being the same.

```
get fillcolor of obj
set fillcolor of obj to ( ( it + 180 ) mod 360 )
```

Layering Order

The layering order (back to front) is:

- All background objects that are NOT DrawDirect, in their specified layering order
- All foreground objects that are NOT DrawDirect, in their specified layering order
- All background objects that ARE DrawDirect, in their specified layering order
- All foreground objects that ARE DrawDirect, in their specified layering order

Modifying Group Components

When modifying an element in a group, there is an annoying problem. That is we have to ungroup the group but afterwards the group name and script are lost. Is there any way or tools or scripts which can help to avoid this problem, or at least to keep the name and the script of a group when modification is necessary?

As long as you don't need to either re-order or delete inside a group, you can manipulate the properties of any object in a group. Say you want to move a rectangle "box" inside a group "house" to the left. You can issue this from the command window:

```
decrement item 1 of position of rectangle "box" of group "house" by 15
```

On a VGA, that will move the box 1 pixel left without changing anything else in the group. Booklook is often a very valuable tool for doing this task.

An example of a script that could go in a sysBook to given below which would allow group components to be altered:

```
to handle rightbuttondown isShift, isCtrl
  conditions
    when isShift is FALSE and isCtrl is FALSE
      edit script of target
    when isShift is TRUE and isCtrl is FALSE
      edit parent of target
    when isShft is TRUE and isCtrl is TRUE
      edit parent of the parent of the target
  end conditions
end
```

You don't want to ungroup if you can avoid it.

I need to "depth arrange" objects that are in a group (from within a script). The "layer" value seems to always be 1, and changing it has no effect. I don't seem to be able to select the object so I can "send bringCloser" or whatever. I would value any ideas or suggestions you may have.

Can't be done! However, you can do it by other means:

Store all the properties of the group in variables - Store the objects of the group in a variable (call it vObjSel) - Ungroup - re-layer - set selection to vObjSel - group - reset all the properties

AVI PlayBack Window

The following script shows how to place an avi window

Place this in your enterbook handler in your book script:

```
-- *** for video display ***
hide rectangle "myAVIBoundingBox" of this page
linkDLL "user"
    -- this function is used to move a window
    INT MoveWindow (WORD, INT, INT, INT, INT, INT)
end
linkDLL "tbkwin.dll"
    -- this function returns coordinates in screen units
    -- it takes coordinates in page units as a parameter
    -- this is used to set the position of Popup and Overlapped
    -- windows
STRING screenFromPage (WORD, STRING, INT, STRING)
    -- this function returns coordinates in client units
    -- it takes coordinates in page units as a parameter
    -- this is used to set the position of Child windows
STRING clientFromPage (WORD, STRING, INT, STRING)
end
-- ***
```

Place this in a button script, or other suitable place:

```
to handle buttonDown
get my fileName
if it is null
    send rightButtonUp
end
set syscursor to 4
    -- close any currently playing AVI files
get tbkMCI("close AVIfile","")
    -- open the AVI file and get its window's handle
get tbkMCIchk("open" && my fileName && "alias AVIfile style
popup","",1,1)
    -- replace the previous line with this line for a child window
    -- get tbkMCIchk("open" && my fileName && "alias AVIfile style child","",1,1)
set hWin to tbkMCIchk("status AVIfile window handle","",1,1)
```



```

-- Determine the coordinates of the new window to move the video to
set wRect to screenFromPage(sysWindowHandle, sysPageScroll,\
  sysMagnification, bounds of rectangle "myAVIBoundingRectangle"\
of this page)
-- replace the previous line with this line for a child window
-- set wRect to clientFromPage(sysWindowHandle, sysPageScroll,\
--   sysMagnification, bounds of rectangle, "myAVIBoundingRectangle"\
--   of this page)

-- Break coordinates up into x,y,width,height
set x to item 1 of wRect
set y to item 2 of wRect
set nWidth to item 3 of wRect - item 1 of wRect
set nHeight to item 4 of wRect - item 2 of wRect

-- move the video window to its new position
get MoveWindow(hWin,x,y,nWidth,nHeight,1)

-- show the video in its new window position and play it
get tbkMCI("window AVIfile state show","")
get tbkMCI("play AVIfile wait","")
get tbkMCI("close AVIfile","")
set sysCursor to 1
end

to handle rightButtonUp
  get OpenDlg(".", "*.avi", "Choose an animation file\
    for this button.,"Choose Animation File")
  if it is null
    clear my fileName
    break to system
  else
    set my fileName to it
  end
end
end

```

This also requires that you draw a rectangle of the size and position you want the video to appear, and call it "myAVIBoundingRectangle". When you right-click the button, you can choose the appropriate video file.

Executing Multimedia Event Without Locking Screen

How can I execute a multimedia event without locking the screen using a WAIT statement?

Example:

```
to handle enterPage
  do whatever
  get tbkMCI(...)
```

-- be sure and specify the page as a target for the message tbkMMNotify

```
to handle tbkMMNotify
  go to next page
end
```

TbkMMNotify is a message that is sent to the specified object when a multimedia event is successfully completed. This specification is one more argument in the tbkMCI call that you are making. See the widget book for exact syntax.

Compare this option with the "wait" option and you realize that whereas "wait" locks up the system, tbkMMNotify allows you to continue execution of OpenScript commands (show field etc.)

Playing Sound Through Non-MCI Device

*I want one windows application (Toolbook version 1.5 *not* a multimedia version) to play a .wav file). Is this possible?*

Use the multimedia extensions *in windows* to do this:

```
linkdll "mmsystem.dll"  
    int sndPlaySound (string,int)  
end  
set vSoundFile to "ding.wav"  
get sndPlaySound (vSoundFile, 0)
```

OR

There is a utility available called WAVER. If it is running a doubleclick at a sound file will play it instead of just loading it. A side effect is that you can play a WAV file from toolbook by simply running the wav file like:

```
to handle playsound  
    run"soundname.wav"  
end
```

Stopping An Animation

There are two ways I know of to write an animation which can be stopped:

1. Drive the animation via an idle handler which increments position values, etc. This means that some of the information will probably have to be stored in system variables or user properties. A sample toolbook employing this type of logic is the "flow.tbk" available on the toolbook BBS.
2. In a long script, sprinkle in calls to the "keystate()" function and break out of the handler if a certain key (or mouse button) is press. Keystate() is documented in toolbook openscript help.

Further comments to above: you may want to filter the mouse position to make sure it is in the neighborhood of the stop button rather than random clicks from nervous users. Possibly use `sysmouseposition` and break the script if the position falls within the bounds of the stop button. We have even "shown" a bitmapped cover for the stop button to make it look like the button is depressed so the user knows an action was taken (since the normal pressing of the button won't take place until the script is interrupted). You may also want to incorporate the `eatcliks` script so additional button presses that might occur while the animation script is waiting to be broken won't take effect after the script is stopped.

Multimedia ToolBook

Does having MM Toolbook eliminate the need for MMDK?

Yes. MM ToolBook = Regular ToolBook + MMDK.

Printing Bitmaps

There is no facility for printing a bitmap that is being played through tbkbmp.dll.

On the page, background, or book you need to capture the creation of the bitmap so you can move it, etc:

```
to handle make
  system svFileName
  if object of target is "PaintObject"
    set name of object to svFileName
    set bounds of PaintObject svFilename to bounds of rectangle "frame"
    -- you'll draw this rectangle as part of your screen design
  end
end
```

Now you can play with the bitmap as a ToolBook object. You'll need to clear it when you are done, and if you allow saves, you'll want to save as not to keep the space for the bitmap around.

Printing Text Of Ordinary Field

You can print plain text to the default font on your printer with the following:

```
to handle printText
  set printFile to "lpt1"                --use standard printer name
  set pageFeed to ansiToChar(12) --page feed for end of file
  openfile printFile
  get text of field "whatever"
  writefile it to printFile
  writefile pageFeed to printFile
  closefile printFile
end printText
```

This will lose any text formatting and font changes however.

OR

The best way to print (which requires a little work) in ToolBook involves a hidden recordField in your background. If you have to use fields in the foreground and you want this text printed then this is the best strategy.

Here's how to do it: Create a recordField in your background and position it so the user can't see it. You can put this outside of the screen area or underneath a foreground object. For this example, we will give it the name "printText".

I assume that you have a field in your foreground. For this example, we will call it "data".

Let's then create a button called "print". In this button we will put the following script:

```
to handle buttonUp
  set sysCursor to 4
  set sysLockScreen to TRUE
  select text of field "data"
  send copy
  set focus to recordField "printText" of this page
  send paste
  send print
  set sysLockScreen to FALSE
  set sysCursor to 2
end
```

```
to handle print
  set printerStyle to groups
  set printerGroupsAccross to 1
  set printerFieldNames to FALSE
  set printerFields to printText
  set printerMargins to 1440,1440,1440,1440
  set printerFieldWidths to 8640
  set printerClipText to FALSE
  set printerBorders to FALSE
  set printerArrangement to 1,1
  start spooler
    print
  end spooler
end
```


Printing Text In Fields In Multiple Pages

Make sure that the fields you want to print all have some property (like pPrintField) set, and make sure that all of these fields have meaningful names. I'd probably write to a file and print the file in Word, but it's up to you:

```
createfile "output.txt"
step i from 1 to pagecount of this book
  set vObj to objects of page i of this book
  while vObj is not null
    pop vObj
    if object of it is "field"
      if pprintfield is true
        writefile name of it & TAB to "output.txt"
        writefile text of it & CRLF to "output.txt"
      end
    end
  end
end
closefile "output.txt"
```

However, if you want direct output you can have it sent to the printer directly.

To send output directly to your printer name the output file lpt1.dos (for a printer hooked to lpt1 of course) and dos will send it to your LPT1 port. For other ports insert the appropriate name of the port followed by any extension you wish.

Playing Sound

Is it possible to loop a .wav file to keep playing while the reader stays on the particular page that is associated to the .wav file?

```
to handle enterpage
  send tbkmmnotify
end

to handle tbkmmnotify
  get tbkmcI ("play foo.wav",self)
end

to handle leavepage
  get tbkmcI ("close foo.wav","")
end
```

How can I play a .wav file and not have to wait until that .wav file has finished playing?

This is the script for playing a sound file, and it assumes that no other MCI event is occurring for the particular page (substitute your directory and file name):

```
to handle enterPage (or buttonup, etc)
  get tbkMCIchk ("play c:\winvideo\intro.wav","",1)
  get yieldApp()
end
```

If a "wait" message is used everything will be inoperable until the particular event is finished.

How can I play WAVs and also flip pages

Place the following 2 handlers in a button to play the file (remember to change the filename to one that appears on your system):

```
to handle buttonUp
  set filename to "c:\sound\sample.wav"
  get tbkMCIchk("open"&&filename&&"alias wav","",1,1)
  get tbkMCIchk("play wav",self,1,1)
end
```

```
to handle tbkMMNotify
  get tbkMCI("close wav","")
end
```

place this handler in the script of the page

```
to handle leavePage
  get tbkMCI("close wav","")
end
```

OR

You want to use the "play and notify" versus "play and wait", the difference being the "wait" parameter.

Place this in the book script (naturally, this assumes that tbkmm.sbk is in your startupSysBooks):

```
to handle enterBook
  get tbkMCI("open foo.wav alias waveFile","") -- where foo.wav is your file
  forward
end
```

Place this in the script of the opening page:

```
to handle enterPage
  forward
  get tbkMCI("play waveFile",self)
end
```

```
to handle tbkmmNotify
  get tbkMCI("status waveFile mode","")
  if it contains "stop" or it contains "pause"
    get tbkMCI("close waveFile","")
  end
end
```

tbkMMNotify is sent whenever a tbkMCIchk() or tbkMCI() call requests it. If you look at page 43 of "Using Multimedia ToolBook" you will see an explanation of the second parameter to the above mentioned functions. This parameter is the <object identifier>, which is the object to notify when the requested operation is complete. If you don't specify a notify object, then tbkMMNotify is not sent.

Changing Directory

Does anyone know of a simple way from within ToolBook to change Window's current directory?

Use tbkFile.DLL function setCurrentDirectory; both the linking of tbkFile.DLL and the function are described in Openscript.hlp. If you use a button to change the current directory the following script should work:

```
to handle buttonup
  linkdll "tbkfile.dll" -- this file must be in your current directory
    INT setCurrentDirectory(STRING)
  end linkDLL
  get setCurrentDirectory("C:\MYDIR") -- needs drive letter !!!
-- you can check It now for errors: 1=OK -1=ErrorOccurred
  unlinkdll "Tbkfile.dll"
end
```

Note that setCurrentDirectory NEEDS the drive letter, but getCurrentDirectory("C") will not give it back. ie:

```
getCurrentDirectory("C") -- this will set It = "MYDIR" note: no "\" either!!
```

This can be a pain because to get the current working dir, you actually have to do this:

```
set currWkDir to (getCurrentDrive() & ":\\" & getCurrentDirectory)
```

ASK And REQUEST Dialog Boxes

In my current project I have some dialog boxes pop up on the screen using the command "ask". However, the default location of these boxes is not satisfactory, and I would like to change it. Will Toolbook allow me to do this, or will I have to do some Windows programming to get the control I desire?

Ask and Request do not let you set the position of their dialogs. However, you can use `tbkdlg.dll` and `dialog.tbk` to create dialogs and position them--without doing windows coding.

Sorting Text in ListBox

I am trying to use the tbkdlg.dll to produce a dialog box that has a listbox in it. Simple so far. I also have no problem getting the initial values defined in the listbox through the dialog.tbk utility. What I want to do is load the values in the listbox of the dialog box at run time. This also works with just one minor aggravation. It puts a blank line at the beginning of the listbox followed by the list I intended to be there. How come?

There is a bug in the listbox code that interprets your last CRLF as a new entry. So to avoid display, simply peel off the last two chars of the variable before passing it to setvalue

Using Icons In Dialog Boxes

You can't embed icons in dialog boxes with toolbook, as `tbkdlg.dll`, which handles dialog creation, only contains certain specific icons. However, this can be circumvented if you don't mind altering your source code.

Example: Using the Visual C resource editor, you may add an icon to `tbkdlg.dll`. This icon may be imported or created with the default pixel editor. Name this icon, and save your new `tbk.dll` under the same name and location.

Create a dialog box with `dialog.tbk`, using any icon in place of the new one you wish to use. Embed the dialog box into a button that is copied into your application. Edit the `userProperties` of this button. You will have to change `dlgInit` and `dlgBox`, replacing the name of the default icon (for example, `question`) with the name of the icon you have created.

Always On Top

How do I make a ToolBook window be Always On Top?

This feature is available under Windows 3.1, but not 3.0).

When a window is Always On Top, it (or its icon) is always visible, even if other application's windows are maximized or positioned over it. This script will make the ToolBook window always on top, but be careful, it will even be in front of all the dialog boxes.

1. Link the function `setWindowPos()` from the Windows API:

```
linkDLL "user"  
    INT setWindowPos (WORD, INT, INT, INT, INT, INT, WORD)  
end linkDLL
```

2. Call `setWindowPos()` with the following parameters:

```
get setWindowPos (sysWindowHandle, -1, 0, 0, 0, 0, 3)
```

where `sysWindowHandle` is the window handle of the main ToolBook window, `-1` says to make the window topmost (to make it non-topmost, change the `-1` to `-2`), `3` tells it not to size or move the window.

This is the explanation:

`BOOL SetWindowPos (hwnd, hwndInsertAfter, x, y, cx, cy, fuflags)`

- | | | |
|------------------------------|---|---|
| <code>hwnd</code> | - | handle of Window (<code>sysWindowHandle</code> in Toolbook) |
| <code>hwndInsertAfter</code> | - | the Window to precede this window in the Z-Order (<code>-1</code> says make it a top-most window) |
| <code>x,y</code> | - | position of Window (we aren't moving the window, so see the <code>fuflags</code> section and leave these 0) |
| <code>cx,cy</code> | - | width and height of the window (we aren't sizing the window, so see the <code>fuflags</code> section and leave these 0) |
| <code>fuflags</code> | - | we want <code>SWP_NOMOVE</code> (2) and <code>SWP_NOSIZE</code> (1) flags, so this value has to be 3. |
| Return value | - | non-zero indicates success, 0 indicates failure. |

Exiting Windows From Toolbook

There is a function in the USER.EXE system file called, surprisingly enough, ExitWindows and can be accessed by the following declaration:

```
to handle exitWindow
  linkDLL "user"
    INT ExitWindows (DWORD,WORD)
  end
  get ExitWindows(0,0)
  unlinkDLL "user"
end
```

This will exit Windows immediately, without displaying the Program Manager. 'This will end your Windows session, however, when this function is called, Windows starts the shutdown by closing applications. If an application has an unsaved data file, it will follow the normal convention of asking the user to save current changes etc. The only way to prevent this happening is to make sure that all data files are saved.

Removing Windows Styles

Include the following in your book script and send the message from your enterBook handler.

This script available on the Asymetrix BBS.

- The initState and restoreStyle handlers could be called from within ToolBook at any point.
- The initState handler is an example of the syntax you would use in your enterBook handler to
- call the SetStyle handler to remove the System Menu and Minimize and Maximize Boxes

--to handle initState

```
--      send setStyle -8-2-1
--end
```

--to handle restoreStyle

```
--      send setStyle 8+2+1
--end
```

to handle SetStyle pValue

- This handler calls into the Windows 3.x DLL, user.exe,
- to get the style of the ToolBook window, modifies that
- value, then sets the window style to this new value.

```
linkdll "user"
    dword getWindowlong(word,int)      -- (hwnd,nIndex)
    dword setWindowlong(word,int,dword) -- (hwnd,nIndex,wNewWord)
end
set style to getWindowLong(sysWindowHandle,-16)
```

-- Add/Remove a Style Item

- Add or subtract one of the following values to style to add
- or remove the corresponding style from this window.

- (1) Maximize Box
- (2) Minimize Box
- (4) Thick Frame
- (8) System Menu
- (16) Horz Scroll
- (32) Vert Scroll
- (64) Dialog Frame
- (128) Border
- (192) Caption {This is 64+128. Includes border and dialog frame.}

```
-- (256) Maximize
-- (512) Clip Children
-- (1024) Clip Siblings
-- (2048) Disabled
-- (4096) Visible
-- (8192) Minimize
-- (16384) Child
-- (32768) Popup
```

```
set hiWord to style div 65536
set loWord to style mod 65536
increment hiWord by pValue
set style to (hiword * 65536) + (loWord)
get setWindowLong(sysWindowHandle,-16,style)
```

```
-- This properly resets the size of the ToolBook window
```

```
hide mainwindow
set bounds of mainwindow to bounds of mainwindow
send sizeToPage
show mainwindow
```

```
end
```

ToolBook to ToolBook (not DDE)

If the program that you want to control is a ToolBook application, then DDE is not necessary. Just remember what page you're on with a system variable and use the "go" command:

```
to handle buttonUp
  system s_returnPage
  set s_returnPage to this page
  go book <bookname>
end
```

Then to return when you are done:

```
to handle buttonUp
  system s_returnPage
  go s_returnPage
end
```

Just make sure that you're not doing a restore system anywhere and this will work fine.

There is no reason to use DDE between ToolBook books, in fact its a bad idea. Its much faster, and simpler, to do things directly. To operate on book A from within book B, you just use complete references.

example:

```
set x to text of field "foo" of page 1 of book "c:\foobar\B.TBK"
send foo to book "c:\foobar\B.TBK"
```

Think of ToolBook as a database management system and the books as two database files. If you were using dBase, you wouldn't run two instances of dBase just to work on two different files.

Another explanation: Since ToolBook is a rather sophisticated Object-Oriented Database Manager, the properties of any object that exist within in the domain of the process are available to any running process. Thus, if you want to get something from book "book1.tbk" in "book2.tbk", simply ask for it:

```
get word 1 of text of field "plik" of book "book1.tbk"
```

ToolBook to PaintBrush (not DDE)

Paint Brush doesn't support DDE. So here's is the brute force method using some API. Place this code in your Book script and call the last four handlers from your code, they should be self explanatory.

```
to handle enterBook
  forward
  send linkDLLs
end

to handle linkDLLs
  linkDLL "user"
  WORD FindWindow (STRING,DWORD)
  INT ShowWindow (WORD,INT)
  LONG SendMessage (WORD,WORD,WORD,WORD)
end
end

to handle runApp className, exeName
  get FindWindow(className,0)
  if it <> 0
    get ShowWindow(it,1)
  else
    run exeName
  end
end

to handle closeApp className
  get FindWindow(className,0)
  if it <> 0
    get SendMessage(it,16,0,0)
  end
end

to handle runPaintBrush
  send runApp "pbParent","pbrush.exe"
end

to handle closePaintBrush
  send closeApp "pbParent"
end
```


ToolBook to Excel

How can I place an Excel graph in a ToolBook application?

It is reasonably straightforward to set up an Excel spreadsheet and macro to transfer graphs from Excel to ToolBook. I have been able to do this following the scripts in the sample book `exceldde.tbk` that I got from the Asymetrix BBS. The trick is to send the data from the table in ToolBook to Excel by dynamic data exchange. The Excel spreadsheet contains a graph of those cells. You then run a macro in Excel by DDE that copies the graph into the paste buffer. Back in ToolBook, you then just "send paste" and position it over the old graph.

The script checks first that Excel is open. If not, it opens it. In `exceldde.tbk` this is done on two separate buttons: Run Excel Docs and Recalculate.

This is what's on the button "Recalculate":

```
to handle buttonUp
```

```
    -- Make sure that the Excel spreadsheet is open.
```

```
    getRemote "R1C1" application excel topic "exceldde.xls"
```

```
    if it is null or ("Failed" is in sysError)
```

```
        show group "MustPressFirst" of this page
```

```
    else
```

```
        -- While this handler is executing set the cursor to an hourglass
```

```
        set sysCursor to 4
```

```
        -- Do not show the screen updates during this handler
```

```
        set syslockscreen to true
```

```
        clear sysError
```

```
        -- If there is an old chart on the page, clear it from the page.
```

```
        if objects of this page contains "picture"
```

```
            select Picture "DDE-Chart"
```

```
            send clear
```

```
        end
```

```
        -- Set the text of the field "test names" to the 3rd through
```

```
        -- 11th rows of column 1 of "exceldde.xls"
```

```
        setRemote "R3C1:R11C1" to text of field "test names"\  
        application excel topic "exceldde.xls"
```

```
        -- Set the text of the field "test sales" to the 3rd through
```

```
        -- 11th rows of column 2 of "exceldde.xls"
```

```
        setRemote "R3C2:R11C2" to text of field "test sales"\  
        application excel topic "exceldde.xls"
```

```
        -- Run the Excel Macro "pastepix.xlm" which copies the chart
```

```
        -- "exceldde.xls" creates to the clipboard.
```

```

        executeRemote "[Run("""PASTEPIX.xlm!pastepix""")] application excel
topic system
        -- Paste the Excel chart (it was sent as a picture) into this application.
send Paste
set name of selection to "DDE-Chart"
move the selection to 30,3480
set bounds of selection to 3795,3705,9390,6435
        -- Show updated screen
set syslockscreen to false
        -- Reset the cursor to its original shape
set sysCursor to default
end if
end buttonUp

```

Here's what's on the "Run Excel Docs" button:

```

to handle buttonUp
    hide group "MustPressFirst" of this page
        -- Run the Excel file "exceldde.xls" minimized.
run "excel.exe exceldde.xls" minimized
        -- Open the Excel Macro file "pastepix.xlm"
executeRemote "[Open("""pastepix.xlm""",0)]"\
    application excel topic system
        --Set a user defined property
end buttonUp

```

The following scripts were provided in a posting to the ToolBook listserv as an example of scripting used to initiate a DDE session between ToolBook and Excel.

BOOK SCRIPT

```

to handle enterbook
    linkdll "kernel"
        WORD winopenfile=openfile(String, POINTER, WORD)
        WORD globalAlloc(WORD,DWORD)
        WORD globalFree(WORD)
        POINTER globalLock(WORD)
        WORD globalUnlock(WORD)
    end linkdll
    linkDll "user"
        WORD FindWindow(DWORD,STRING)
        INT PostMessage(WORD, WORD, WORD, LONG)

```



```
        WORD SetActiveWindow(WORD)
    end linkdll
end enterbook

to handle leavebook
    send buttonup to button "quit" of page 1 of this book
end leavebook
```

PAGE SCRIPT

```
to handle enterpage
    system excelxlsfile,num
    ask "Name of excel file?"
    set excelXLSFile to it
    get "c:\excel\excel.exe" && excelXLSFile
    run it
    get findwindow(0,"Microsoft Excel")
    set num to it
end
```

BUTTON "QUIT"

```
to handle buttonup
    system num
    get showwindow(num,1)
    get setactivewindow(num)
    get postmessage(num,16,0,0)
end buttonup
```

EXAMPLES OF EXCEL DDE MESSAGES

GETREMOTE

```
getRemote systems application excel topic system
getRemote topics application excel topic system
getRemote status application excel topic system
getRemote selection application excel topic system
GetRemote "R4C3" application excel topic excelxlsfile
getRemote "R4C1:R4C5" application excel topic excelxlsfile
getRemote cell application excel topic excelXLSfile
--cell defined elsewhere
```

EXECUTEREMOTE

```
executeRemote ":CLOSE(TRUE):" application excel topic excelxlsfile
  ":SELECT("R3C4"): "
executeremote it application excel topic excelxlsfile
  ---it defined elsewhere
":select("c2"): " -- select a column
":select("RC:1:"): " -- increment column
":select("R:-2:C:1:"): " -- decrement row by 2
":formula.find(4.3,1,1,1): " -- find 4.3 in formulas, whole, by rows
-- the following sequence selects cells, constructs a chart,
-- and closes it
":select("R2C4:R5C5"): "
":new(2): "
":close(): "
--
":OPEN("DDECHART.XLC",0,0): "
":PAGE.SETUP("A page","Page &p",0.75,0.75,1,1,TRUE,FALSE): "
":show.info(): "
```

SETREMOTE

```
setRemote cell to val application excel topic excelxlsfile
  --cell, val defined elsewhere
```

ToolBook to Access

-- make sure Access is running

```
keepRemote application msaccess topic "DatabaseName;SQL Select * from  
TableName"
```

```
getRemote "firstrow" application msaccess topic "DatabaseName;SQL Select *  
from TableName"
```

```
put it
```

```
step i from 1 to 9
```

```
    getRemote "nextrow" application msaccess topic "DatabaseName;SQL Select *  
from TableName"
```

```
    put it after commandwindow
```

```
end
```

```
closeRemote application msaccess topic "DatabaseName;SQL Select * from  
TableName"
```

This puts the contents of the first 10 rows of the specified table of the specified database into the command window.

ToolBook to Word For Windows

It is often preferable to go after the window classname as it almost never changes:

```
linkdll user
  word FindWindowCaption = FindWindow (long,string)
  word FindWindowClass   = FindWindow (string,long)
end

set vhwndExcel to FindWindowClass("excel",0)
--will find the excel window handle, even when the caption is "Excel - MYSHEET.XLS"
```

Subject: Re: application control

DDE would only work for Word, Paint Brush doesn't support DDE. So here's is the brute force method using some API. Place this code in your Book script and call the last four handlers from your code, they should be self explanatory.

```
to handle enterBook
  forward
  send linkDLLs
end

to handle linkDLLs
  linkDLL "user"
  WORD FindWindow (STRING,DWORD)
  INT ShowWindow (WORD,INT)
  LONG SendMessage (WORD,WORD,WORD,LONG)
end

end

to handle runApp className, exeName
  get FindWindow(className,0)
  if it <> 0
    get ShowWindow(it,1)
  else
    run exeName
  end
end

end

to handle closeApp className
  get FindWindow(className,0)
```

```
    if it <> 0
        get SendMessage(it,16,0,0)
    end
end

to handle runWinword
    send runApp "OpusApp","winword.exe"
end

to handle runPaintBrush
    send runApp "pbParent","pbrush.exe"
end

to handle closeWinword
    send closeApp "OpusApp"
end

to handle closePaintBrush
    send closeApp "pbParent"
end
```

Avoid Launching Active Applications

To avoid launching active applications use the FindWindow function to find window handles by the class name.

```
linkDLL "user"
    WORD FindWindowByClass=FindWindow (STRING,DWORD)
    WORD FindWindowByCaption =FindWindow (DWORD,STRING)
end
```

example:

```
get FindWindowByClass("OpusApp",0)
    -- this will find Winword by the class name, regardless of the caption.
get FindWindowByClass("XLMAIN",0)
    -- this will find Excel by the class name, regardless of the caption.
```

However, FindWindow will only find a window if it knows the exact caption.

There are several appropriate windows api calls to solve this problem.

```
GetNextWindow() or EnumWindows() and EnumWindowsProc()
GetClassName()
```

The following is a pseudo code example of how to find out if a program is already running.

```
--linking the dll functions must have already occurred.
set starthandle to syswindowhandle
set temphandle to starthandle
set done to FALSE
do
    set temphandle to getnextwindow(temphandle)
    if getClassName(temphandle) = nameofProgramtorun
        set done to true
    end if
until done = true or temphandle = starthandle
if done = false
    run nameofProgramtorun
else
    request "already running."
end if
```

Sizing Other Applications From ToolBook

Here is a method to resize the windows of other applications. To do this you have to get the window handle of the other applications. Most of the time you can obtain this with a call to the function FindWindow as in this example. Once you have the handle, use the function SetWindowPos to set the position and size of the window. Finally, to get the ToolBook window back to the front, use the SetActiveWindow function.

```
to handle buttonUp
    linkDLL "user"
        word FindWindow(string,dword)
        int SetWindowPos(word,word,int,int,int,int,word)
        word SetActiveWindow(word)
    end
```

--OpusApp is the window class name of Word For Windows

```
set hWordWnd to FindWindow("OpusApp",0)
```

--Use SetWindowPos to position the window and resize

```
--SetWindowPos(hWnd,zOrder,x,y,width,height,flag)
```

- hWnd = handle of window you want to resize
- zOrder = used to adjust z order, set it to 0
- x = position of upper left hand corner of window in pixels
- y = position of upper left hand corner of window in pixels
- width = width of window
- height = height of window
- repaint flag, set to 4 to ignore zOrder flag above

```
get SetWindowPos(hWordWnd,0, 0,0,640,480 ,4)
```

```
get SetActiveWindow(sysWindowHandle)
```

```
end
```

Password Protection

There is no function to decrypt the password. However it is discoverable in the *simplest* of all cryptographic analyses.

Author's Note: I would strongly recommend that you do not use password protection for your book scripts. The algorithm used to encrypt passwords is extremely simple (it took me 5 minutes to work it out). If you really want to protect your scripts then I would suggest joining the [Developers' Service](#) and obtaining the script remover utility.

Determining If Other Applications Are Running

How can I find out if certain ToolBook programs or Windows programs are running, and if so, bring the running instance to the front?

-- link the DLL's that we'll need

```
to handle LinkDLLs
  set sysSuspend to false
  linkDLL "kernel"
    int getModuleFilename(word,string,int)
  end
  linkDLL "user"
    int bringWindowToTop(word)
    int IsIconic(word)
    int OpenIcon(word)
    word getWindow(word,word)
    word GetClassWord(word,int)
  end
  set sysSuspend to true
end LinkDLLs
```

-- Find out if an EXE file is running and if so, what is its window handle.

```
to get HandleOfExeFile ExeFile
-- get first window in the window manager's list
  set wh to GetWindow(sysWindowhandle,0) -- 0 returns the first window
-- Step through all remaining windows
  do
    if uppercase(ExeFile) is in NameOfFile(wh)
      return wh
    end if
  -- Get the next window handle in the window manager's list.
  -- The parameter value "2" causes getWindow to return either
  -- the handle of the next window in the list or "0" if you
  -- are already on the last window.
    set wh to getWindow(wh,2)
  until wh = 0
  return wh -- 0, i.e. ExeFile is not running
end HandleOfExeFile
```

```
to get nameOfFile wh
```

-- get the moduleHandle for the window.

```

    set mh to getClassWord(wh,-16)
-- initialize buffer of 40 spaces so we don't blow memory.
    set buffer to "                                     "
    get GetModuleFilename(mh,buffer,41) -- 40 + null termination
    return buffer
end nameOfFile

-- Example: You want the student to click a button in your book in
-- order to run WRITE.EXE so he write an essay.
-- You set a system variable to sEssayProgram somewhere (it could be
-- WRITE.EXE or WORD.EXE etc.)
-- If sEssayProgram is already running, and perhaps is minimized,
-- you want it to be restored and brought to front. If not, start it from
-- some directory in the Path, or from your program's home directory.
-- Note that sPath is a system variable set to your program's home directory

to handle runEssayProgram
    system sEssayProgram, sPath
    local lCursor, sEssayHandle
    set lCursor to sysCursor
    set sysCursor to 4
    set sysSuspend to FALSE
    set sEssayHandle to HandleOfExeFile(sEssayProgram)
    if sEssayHandle <> 0 -- it is running already
        if IsIconic(sEssayHandle) <> 0
            get OpenIcon(sEssayHandle)
        end if
        get bringWindowToTop(sEssayHandle)
        set sysSuspend to TRUE
        set sysCursor to lCursor
        break
    end if
-- if it is not running:
    clear sysError
-- first try some PATH directory
    run sEssayProgram
    if sysError is not null
        clear sysError
-- try running from home directory
        run (sPath&sEssayProgram)
        if sysError is not null
            request "Program," && sEssayProgram && "cannot be started."

```

```

        set sysSyspend to TRUE
        set sysCursor to lCursor
        break
    end if
end if
set sysCursor to lCursor
set sysSyspend to TRUE
end runEssayProgram

```

-- when I you want to find out if a ToolBook app is running, I use this:

```

to get isBookRunning bookName
    clear sysError
    getRemote "this book" application toolbook topic bookName
    if "OK" is in sysError
        return TRUE
    else
        return FALSE
    end
end
end

```

-- Walter Knowles suggests setting up a true global variable as a flag that
-- says, "I'm running", when your book starts up, and clearing this
-- flag when it is closed:
-- You must have mm toolbook, and add tbkmm.sbk to startup sysbooks

```

to handle enterBook
    forward
-- g_open is your flag; it can be set to true/false, or to a name of a specific book
-- test the flag: was it set by an earlier start-up?
    get getGlobalVar(g_open)
    if it is NULL
-- set the flag to TRUE
        get setGlobalVar(g_open, TRUE)
    else
-- the flag has already been set
        request "This book is already open."
        send exit
        linkDLL "user"
            int bringWindowToTop(word)
        end linkDLL
-- this will bring the first ToolBook app in the list to front:

```

```
        getRemote "sysWindowHandle" application toolbook
        get bringWindowToTop(it)
-- if you want a specific ToolBook app, add 'topic' to getRemote
    end if
end enterBook

to handle leaveBook
    get setGlobalVar(g_open, NULL)
    forward
end leaveBook
```

Deactivating Button

Three things need to be done to deactivate a button.

1. Set "highlight" of that button to false
2. Set "excludeTab" of that button to true
3. Set strokeColor of that button to some shade of grey (0,50,0)

The button should check itself when it is clicked to find out if one of the above conditions is not met - then it should execute its script i.e.,:

```
to handle buttonUp
  if highlight of self = FALSE
    -- do something
  end if
end
```

I use a function on the book script since it is cumbersome to write these three lines over and over again. Pass a valid object identifier for the button to be activated/deactivated along with a flag to indicate which setting is desired. i.e.;

```
get setButton(button "doSomething",1)
```

There is sample code designed to do this in TEMPLATE.ZIP, which should be on the Asymetrix ftp site.

Conversion Of HyperCard Stacks To ToolBook Books

There is a program called ConvertIt! for converting HC stacks to TB books from Heizer software that does an 80% job. There is no reverse path.

Heizer Software
P.O. Box 232019
Pleasant Hills, CA 94523

VOICE: 800-888-7667
FAX: 415-943-6882

ConvertIt! is a useful tool when used for a limited purpose, that is, to convert HyperCard's basic elements (backgrounds, cards, fields, buttons, text, etc.). Everything is placed properly on the pages in ToolBook, including the specific fonts that you define as part of the conversion process. Fields and buttons had comparable properties. Graphics also convert quite well.

With respect to the conversion of scripts, ConvertIt! does an excellent job at converting the scripts that can be converted. The lines that can't are intelligently commented so you can go back and figure out the problems. The print buttons do need to be re-scripted.

Of course, any XCMDs (external commands) that a HyperCard stack uses will not be converted and you will need to write your own custom DLLs (dynamic link libraries) in their place.

Debug Window

Disappearing Variable window

This can be solved by reentering the win.ini information for the debug window for Toolbook.

Try something to the effect of:

DebugWindow=9,-13,629,241,306,302,251,363
under the [Toolbook] section of win.ini

Setting debug breakpoints

You set a break point by going into the debugger and clicking on the statement you want to stop on. Break points are instance specific, so you can't set a break point in a book and then close it, because the break point is not saved.

To start the debugger when first run the book, hold down both shift keys (Shift+Shift).

Determining Runtime Or Full Version ToolBook

The system variable sysRunTime is set to TRUE when running under the runtime version of ToolBook. In a book script put the following:

```
get sysRunTime
if it is TRUE
  ....
else
  ....
end
```


Gang Screen

In TBOOK.EXE and TOOLBOOK.EXE there are five 16-colour bitmaps - a male Egyptian in two poses, a female Egyptian in two poses, and a pyramid. To get these bitmaps to show as an animation there is a sequence of mouse clicks on either side of the ToolBook logo in the "About ToolBook" dialog box. Specifically, right+double+click on the right side of the logo, then left+double+click on the left side of the logo.

Limitations

You can have a maximum of 64k of data on a page. That includes ALL text, scripts, userproperties and objects. On a blank page that works out to about 800 rectangles.

A field can contain 32k of text.

The theoretical limit for pages + backgrounds is about < 60000. The practical limit is about 41000.

Everything on a page (except for pictures and bitmaps) is stored in one data segment (let's call it the "page segment"). Properties are stored as arrays of strings (and some other stuff, like hash tables, etc) . Objects take up about 200 bytes for their basic existence, and the page itself takes up about the same amount. So your limit is (approx)

no of objects * 200 bytes +
sum(object built in properties data <like text>) +
sum(charcount (property name) + charcount(properties text) + 2 <for '\0'+
no of userproperties * ~10 +
1K <general overhead fudge factor>

must be less than 64K

Note: this is all experimental data, so it's all approximate.

BTW UserProperties are for authors; they should be considered instance variables of the objects (which they are) . Programs should not willy-nilly create UserProperties.

OTHER RESPONSES:

A ToolBook page can hold up to 65,536 bytes. The average object uses about 60 bytes. A field with text in it uses more space, because each character is a byte, whereas a line uses less space because it is the simplest of all objects. On average, you can get about 1000 objects onto a page, depending on the object types involved. (Technically, the text of a field is moved to a separate data store once the number of characters in the field reaches 2,000, ie, a field with more than 2,000 characters in its text uses up less page space than a field with less than 2,000 characters.)

ToolBook stores bitmaps in two different ways. Small bitmaps (something less than 2K of bits, or about 100x100 monochrome, 50x50 color) are stored as part of the page. Larger bitmaps are stored in a different area of the file, and you only have to pay the object overhead.

Network

The ONLY way to share ToolBook applications on a network is to make the books read only and ensure that TBKNET.EXE is in the local path (preferably on the local machine). Only one user at a time can access a book that is not read only on the network (TBKNET is required for this too).

What is TBKNET.EXE?

ToolBook, on a local machine without TBKNET, opens and closes the OBJ file (book) at will. This is a problem when you're on a net. TBKNET is basically the mechanism used to keep the file open on the net all of the time. When TBKNET is loaded, TBKBASE.DLL (the DBMS) uses it to access the file. In other words, TBKBASE opens the file through TBKNET instead of explicitly opening it.

Runtime Distribution

The licence agreement that comes with ToolBook says that you can copy and distribute the runtime files, i.e. TBOOK.EXE & DLL's with your application. There are no royalties or license fees involved in the distribution of product.

Asymetrix's runtime module will show a first screen that gives them credit. If you belong to their Developer Services you can get software that will allow you to eliminate the first credit screen.

The file that came with your ToolBook product, TBOOK.EXE is the ToolBook runtime file. You still need the proper DLL's to make it work. Take a look at the file RUNTIME.BAT to see what files are necessary for use as the runtime of ToolBook. The runtime does not allow the setting of scripts or the user to go to author mode. There are also a few commands that it does not allow (these are documented in the release notes).

ToolBook Versions

Current ToolBook Version

1.53 is the current version of ToolBook. The MMRK is an add-on product that will make your copy of ToolBook into "Multimedia ToolBook".

Version 2.0

This version is now in final beta (May 1993) and the beta is supposed to be in "several" waves. Since it is much easier to do thorough beta testing rather than reissue an update, I would put my money on 4-5 months before it actually is on the street. However, publically, Asymetrix have not stated a firm release date.

Developers' Utilities

Asymetrix offers some developers utilities and support for a yearly subscription of \$595. Some of the utilities are BookLook which allows you to look at the various type of properties in your book. Searcher which allows you to search for just about anything in your book. Remover which allows you to visually remove your script from your book, the process is transparent because your book works as if the script is there however if a hacker should try to edit your book he/she won't be able to see the script. There is also a simple install utility for automating the installation of a Toolbook application that you distribute.

There is a very handy script called "PageLook" that comes in a package called "ToolBox for ToolBook" by the Redmon Group - available as freeware. The "EatClicks" script is also in this book.

IATDEV.SBK is a system book that provides various reporting utilities and a way to click on an object in Reader mode and see its script.

Both are available from ftp.cica.indiana.edu in pub/pc/win3/toolbook.

You can write a simple script yourself to view the scripts of objects from reader level. Just put something like the following in a sysBook

```
to handle rightButtonUp mLoc, isShift, isCtrl
  conditions
    when isShift
      edit the script of this background
    when isCtrl
      if parent of target is "group"
        edit script of parent of target
      else
        edit script of this book
      end if
    when true
      edit the script of target
  end
end rightButtonUp
```

Something like that works nicely ... If you click on the right mouse button, and the SHIFT key is down, you can see the background script. If just the CTRL key is down, you can edit group scripts at reader level and the book script provided you don't click on a group when you hold down the CTRL key. Since the page can be a target if you just make sure that you don't click on a foreground object and don't hold down SHIFT or CTRL you can edit the page script at reader level.

Setting Timers

because I don't want to have to keep switching time formats), I don't tend to use long idle/systemtime combinations. There is a Windows API:

```
linkdll "kernel"  
    long GetTickCount()  
end
```

and I call that function (which returns a millisecond value). That way I don't worry about it.

> there a way of having TB subtract these values. For my TB books I just set
> the counter up using the tickcount but when I jump to DOS with this button the
> counter stops.

to handle timeExternal

```
    local start_time, end_time, run_time  
  
    set start_time to sysTime  
    run "external"  
    set end_time to sysTime  
    format time start_time as "seconds"  
    format time end_time as "seconds"  
    set run_time to end_time-start_time  
    format time run_time as "min"           -- may be wrong format code  
    set text of field "howLongItTook" to run_time  
end timeExternal
```

Try something along those lines. I doubt that script will work unaltered, but you never know.

I would like to have minutes. I know enough to stamp systime going and coming back in but is there a way of having ToolBook subtract these values.

```
set sysTimeFormat to "seconds"
```

Then when you take a copy of sysTime you have a number (the number of seconds which have elapsed since midnight that day) which you can manipulate - subtract the going from the coming then divide it by 60 if you want minutes.

If for some reason you wanted sysTimeFormat left at "h:min:sec" (the default), then when

you take a copy of sysTime, convert it to seconds with this statement:

```
format time myTime as "seconds" from "h:min:sec"
```

then manipulate it as you want.

There is a useful script in logbook.tbk from the Asymetrix BBS to help with timing student use, viz;

```
to get ElapsedTime ElapsedTimeSecs
  put ElapsedTimeSecs div 3600 into Myhours
  put ElapsedTimeSecs mod 3600 into MyMinSecs
  put MyMinSecs div 60 into MyMins
  put MyMinSecs mod 60 into MySecs
  put Myhours && "hours" && MyMins && "minutes" && MySecs && "seconds" into
TimeString
  return TimeString
end ElapsedTime
```

"mod" is an arithmetic operator that divides the number on its left by the number on its right, yielding only the remainder.

I'm told that there is a timer service in tbkmm.dll, and if you have Multimedia ToolBook, that's probably the easiest route. You can, of course, just call the same functions it does down in windows:

```
translatewindowmessage for sysWindowHandle
  after 275 send alarm
end
linkdll "kernel"
  word SetTimer (word,word,word,dword)
  int KillTimer (word,word)
end
set vTimerMilliSeconds to 1000 -- for a 1-second clock pulse
set vTimerID to 99 -- any int will do
get SetTimer (sysWindowHandle,vTimerID, vTimerMilliSeconds, 0)
```

and after you are done with the timer:

```
get KillTimer(sysWindowHandle,vTimerID)
```

```
to handle enterBook
  system s_lastTime, s_startTime
  linkDLL "user"
```



```
        DWORD    GetTickCount()
    end
    set s_lastTime to GetTickCount()
    set s_startTime to GetTickCount()
    forward
end

to handle idle
    system s_lastTime, s_startTime, s_mode
    local currentTime
    set currentTime to GetTickCount()
    if currentTime - s_lastTime > 500 -- the users has been interacting
        set s_startTime to currentTime
    else
        if currentTime - s_startTime > 300000 -- 5 minutes of inactivity
            set s_mode to "slideShow"
            send startSlideShow
        end
    end
end
set s_lastTime to currentTime
end
```

ToolBook to Calculator

To Get The Calculator:

```
linkdll user
  word FindWindow (dword, string)
  int BringWindowToTop (word)
end linkdll
```

Get FindWindow(0, "Calculator") --must be the exact caption of the app

```
if it=0
  run "calc.exe"
  if syserror is not null
    request "Unable to find requested file"
  end if
end if
```

Get BringWindowToTop(it) --it being the hwnd of Calculator

To Close Calculator:

```
linkdll user
  word FindWindow(dword, string)
  word SendMessage(word, int, word, word)
end linkdll
set hwnd to FindWindow (0, Calculator) --- get the calculator's hwnd
get SendMessage(hwnd, 16, 0, 0) ---send WM_CLOSE (16) message to calculator
```

NOTE: When the Calculator is brought into the Toolbook window, it becomes the active window and can be used normally; clicking OUTSIDE the calculator dismisses it (it is sent behind the toolbook window) and Toolbook is re-activated. Calling it again simply brings it back to the top level. The Closecalculator handler is used only at the end to close the Calc.exe file for good.

tbkBitmap()

Though the "Using Multimedia Toolbook" guide implies that `tbkBitmap()` works only with DIB files, it seems not to mind plain old BMP files.

Confirmed Bug in `tbkBitmap()`

A scanned image ended up in a directory named "pstyler" (Aldus Photostyler) by mistake, so the `tbkbitmap(OPEN ...)` command was edited to use "c:\pstyler\foo.bmp" rather than the student working directory. `tbkBitmap` complained bitterly about type mismatches and parameters not being right. If the directory is renamed to anything else, `tbkbitmap` functions properly.

The substring "style" in "pstyler" is being confused with the `STYLE` keyword in the `tbkbitmap` command syntax. The parser may not be skipping over the filename when looking for that particular reserved word.

Global Changes To Scripts

Here's a simple handler to allow you to look at every object on every page in a book:

```
to handle setActivate
  step i from 1 to pageCount of this book
-- go through the whole book
  put objects of page i into temp
-- get a list of objects on page i
  while temp <> NULL
-- pop the list until it's empty
    pop temp
-- pull the top item from the list of objects on the page
-- initialize special variable "it" to that item e.g., "field id 3 of page
id 7"
    if it contains "field" -- i.e., it's a text field
      set activated of it to TRUE
    end if
  end while
end step
end setActivate
```

Changing the activated property is specific. Once you have a specific object (based on the conditions you evaluate - if fillColor of it is RED and name of it is "video button" - etc.) then you can do with it as you please. This sort of process can save a lot of time when you have to make changes. To be fancy, you may want to do neat things with the sysCursor so that others who use your script will know that something is happening as they search every page in a large book. Step through backgrounds instead of pages to get to background objects.

Script Tips

When writing a script how does one specify a new set of quote material (a button name, for instance) inside the larger quote context?

The second set of quotes is handled with a "E&".....""E statement.

If the script is too long for one line how can this be continued onto the next line?

Use a "\" at the end of a line to continue onto the next line.

Example:

```
to handle buttonUp
  request "How are you today. " \
    && "I am fine thanks"
end
```

Moving And Positioning An Animation Window

MCI does not provide a facility for positioning and sizing an animation frame window on the screen. Use the following handler. (The handler assumes the existence of a rectangle named "Animation Frame.")

```
to handle moveAnimationWindow
```

```
  linkDLL "user"
    INT moveWindow(WORD, INT, INT, INT, INT, WORD)
  end
  linkDLL "tbkwin.dll"
    STRING screenFromPage(WORD, STRING, INT, STRING)
    STRING clientFromPage(STRING, INT, STRING)
  end
end
```

--For child windows

```
  set wRect to clientFromPage(sysPageScroll, sysMagnification, \
    bounds of rectangle "Animation Frame")
```

--For popup and overlapped windows

```
  set wRect to screenFromPage(sysWindowHandle, sysPageScroll, \
    sysMagnification, bounds of rectangle "Animation Frame")
```

```
  set x to item 1 of wRect
  set y to item 2 of wRect
  set nWidth to item 3 of wRect - item 1 of wRect
  set nHeight to item 4 of wRect - item 2 of wRect
  set hwnd to tbkMCI("status aniFile window handle","")
  get moveWindow(hwnd,x,y,nWidth,nHeight,1)
  get tbkMCI("window aniFile state show","")
  unlinkDLL "user"
  unlinkDLL "tbkwin.dll"
```

```
end
```

ToolBook Demo - TestDrive

The book called Welcome to Multimedia by Linda Tway, MIS Press, 1992 includes the Evaluation Edition of ToolBook on CD-ROM. The book is for a beginner, but is highly recommend for anyone who wants to try out ToolBook. It costs approx. \$24.95 in the U.S.A. With the Evaluation Edition you can work on each book at author level for six hours. You can work on many different books--but you can only work on each book at author level for 6 hours. So if you are working on four books, you can spend 6 hours per book (or 24 hours total). Books developed with the application have no time limitation at reader level. Any books you develop using the Evaluation Edition can be used with the full version of the software without any changes

The Evaluation Edition is not intended for development work but would be useful for teaching students how to use ToolBook.

Selecting Text Lines In A Field

How can I select lines in a field?

As it happens, ToolBook has a function called `textFromPoint()` which identifies the textline of an activated field when you pass it the location.

```
to handle buttonUp loc                --script in activated field
  get item 1 of textFromPoint(loc) --returns textline and char num of field
  if it > 0 then
    put textline it of my text  -- display it or whatever...
  end if
end
```

Also remember that you could make the field a single-select listbox. Then the field's `SelectedTextLines` property will tell you which one was most recently selected.

Returning Book To Default State After User Changes

I want to send a handler to each page of a book to hide specific fields and groups when I close the book. I don't want to clean the page upon entering or leaving it because once a field/group is revealed, I'd like to keep it revealed until the session is over. How can I do this?

You have the function - clean up. The event to prompt it is not enterPage but rather enterBook. Standardize the set up handlers with a name like "getSet" then write an enterBook script like:

```
to handle enterBook
  send getReady
end enterBook

to handle getReady
  step i from 1 to pageCount of this book
    send getSet to page i
  end step
end getReady

to handle getSet
end -- In case a page has no "getSet" handler.
```

Be sure that all "getSet" scripts refer to objects with the added identifier "of self" (e.g., hide field x OF SELF)

The purpose of the "getReady" handler is to isolate the clean up from the rest of what you do on enterBook. That way you can call "getReady" from any point in your book and only reset page displays rather than do other things that do not need to be repeated in the enterBook handler.

Note that the event "to handle leavebook" could be used instead if the cleanup were desired upon closing the TBK. This may be preferable if the getSet handlers required time-consuming chores. It could be preferable to have the user sit through the delay at the end, after having formed a good impression of your ToolBook application. Putting delays up front (e.g. enterbook) may start your user off with a pessimistic attitude.

OR

A more simple solution would be not to allow the user to SAVE the book. If he/she cannot save changes, the next time the program is used the fields will be hidden/visible just the way they were before they were changed. You could do this with a "set SysChangesDB to false" command. Setting sysChangesDB to false has the effect of allowing the user to exit without being prompted to save changes. Unless the SAVE option is also removed from

the menus, the user may still save changes and circumvent your intent.

Most Space Efficient Method To Store Bitmaps

What is the most space efficient way to use a bitmap as a background?

What you might try as part of your release cycle (DO NOT DO THIS ON YOUR ONLY COPY OF THE BOOK):

After you have finished the book, go through each background at author level and set the "store as bitmap" of each background. This will store a copy of the background as a compressed bitmap that will image somewhat faster. Save the book then (AT READER) select the bitmap on each background and clear it. Then do a save as to another file name. Your book will image faster and be a bit smaller.

NOTE: THIS ISN'T REVERSABLE.

Determining If Text Will Fit In One Line In Text Field

How can I work out if a sentence will fit into 1 line of a field?

To check if a sentence fits into 1 line of a field, create a normal wrap field which is 1 line deep. Paste text into this field and use the `textOverflow` function to find out if the text is too long. Alternatively, use courier font in your display field and count the chars.

Determining Which Word Is Selected

How can I find which word is selected?

```
set vt to textfrompoint(loc)           -- the location parameter in mouse msg
set loct1 to item 1 of vt              -- textlines before loc
set locch to item 2 of vt              -- characters before loc in cur. line
put wordcount(textlines 1 to loct1 of text of self)\
  into lwords                          --words before the current line
increment lwords by wordcount(characters 1 to locch \
  of textline loct1 of text of self)    --words on current line before loc
```

OR

```
set lwords to wordcount(textlines 1 to item 1 of textfrompoint(loc)\
  of text of self)+wordcount(characters 1 to item 2 of\
  textfrompoint(loc) of textline item 1 of textfrompoint(loc)\
  of text of self
```

Changing The Field Delimiter In An ASCII File

I have an ASCII tab-delimited file that I want to convert to an ASCII comma-delimited file with double quotation marks (ie., "field1","field2",etc).

There are seven fields in each record and a tab is the first character of each record. There are also double quotation marks in some of the tab-delimited fields which I would like to convert to single quotation marks before replacing the tabs as above and clearing the tab from the start of each record.

I can read part of the file into a buffer and then read each record from the buffer by finding the offset(cr,source). I now what to do the changes to each record as described above and save the file in the new format.

```
to handle convertFile
    system filename,fOutFile
    set fOutFile to "intnt.txt"

    --
    -- Get the size of the file
    --
    linkdll "tbkfile.dll"
        long getfilesize(string)
    end
    set filesize to getfilesize(filename)
    unlinkdll "tbkfile.dll"

    set charsRead to 0
    set maxbuffer to 5000
    set j to 0

    createFile fOutFile

    set vOut to QUOTE -- initial quote to start things out

    while charsRead < filesize

        -- read maxbuffer characters, or amount remaining in file. Keep
        -- track of the number of characters read in charsRead
        set charsToRead to min(maxbuffer,filesize - charsRead)
        readFile filename for charsToRead
        increment charsRead by charsToRead
```

```

set vIn to it
set vCt to charCount(vIn)
step i from 1 to vCt
set vCh to char i of vIn
conditions
  when vCh is CR
    put QUOTE & CR after vOut
  when vCh is TAB
    if vPrevCh is LF --we are at the beginning of a line
      put QUOTE after vOut
    else
      put QUOTE & "," & QUOTE after vOut
    end
  when vCh is QUOTE
    put "'" after vOut
  else
    put vCh after vOut
  end
  set vPrevCh to vCh
end
writeFile vOut to fOutFile
if j = 0
  clear chars 1 to 3 of vOut
  closeFile fOutFile
  createFile fOutFile
  writeFile vOut to fOutFile
  increment j
end
clear vOut
end
closeFile fOutFile
closeFile filename
end

```

Verifying A Page's Existence

Example:

```
if not isPage(page "foo")
    request "That page does not exist"
end
```

```
to get isPage pageRef
    -- Determines if a page exists
    local saveSuspend
    set saveSuspend to sysSuspend
    set sysSuspend to FALSE
    clear sysError
    get page pageRef
    set sysSuspend to saveSuspend
    if sysError <> NULL
        return FALSE
    else
        return TRUE
    end
end
end
```


Prevent Switching Windows Using Alt-Tab

Is there a way to prevent switching active windows (with alt-tab)? We don't want readers to be able to exit the book they are in.

There are three main responses to this question:

1. `_subclass_ToolBook`'s main window to trap `WM_SYSKEYDOWN`, and then just don't call `ToolBook's wndProc` when you get an alt-tab [**author**: more info. on this is needed!].
2. Disabling the "Fast Alt-Tab Switching" in the Desktop portion of Control Panel. Unfortunately this does not disable the task switching; it simply changes the way the screen is painted to indicate that you are switching (i.e. instead of the box in the middle of the screen, you see the outline of the window you are about to switch to).
3. Making `ToolBook` act as the shell program instead of Program Manager. In a computer laboratory situation this solution is not viable if `ToolBook` is only one of many applications that are used.

Note: Alt-Tab is **not** a function of Program Manager; it's part of `DefWndProc` in `User`.

Finding Out A Book's Backgrounds

There are basically two ways, you either step through all the pages in the book, or loop through the id numbers of the backgrounds. The latter is generally faster.

Example:

```
request backgrounds ()

to get backgrounds
  local i, bgCnt, retValue
  set bgCnt to 1
  set i to 0
  set sysSuspend to FALSE
  while bgCnt <= backgroundCount of this book
    clear sysError
    get background id i
    if sysError = NULL
      push i onto retValue
      increment bgCnt
    end
    increment i
  end
  set sysSuspend to TRUE
  return retValue
end
```

OR

```
to get backgrounds
  local currentID, i, retValue
  set currentID to 0
  set sysSuspend to FALSE
  step i from 1 to backgroundCount of this book
  do
    increment currentID
    clear sysError
    get background id currentID
    until sysError=NULL
    push currentID onto retValue
  end step
```

```
    set sysSuspend to TRUE
    return retValue
end backgrounds
```

Activate / Deactivate MenuItem

If one builds a drop down menu, is there any way of switching the item choices off depending on which page in the toolbook you are at?

Use activate menuItem & deactivate menuItem.

You could set these as you leave and enter the applicable pages, or you could go for something a little more elegant like setting them everytime the menu is dropped:

```
to handle enterBook
  translateWindowMessage before 278 send WM_INITMENU
  forward
end

to handle WM_INITMENU
  conditions
  when name of this page = "foo"
    deactivate menuItem "foo" at reader
    activate menuItem "bar" at reader
    activate menuItem "foobar" at reader
  when name of this page = "bar"
    activate menuItem "foo" at reader
    deactivate menuItem "bar" at reader
    activate menuItem "foobar" at reader
  when name of this page = "foobar"
    activate menuItem "foo" at reader
    activate menuItem "bar" at reader
    deactivate menuItem "foobar" at reader
  end
end
```

Enable / Disable Window

I have 2 books running. I pass control between them - no problem. The client sends a remoteCommand to the server. The server has the window handle of the client and hides the client window, shows itself, and makes itself active. When the server wants to return control to the client, it shows the client window, makes the client active, and hides itself. I now want to keep the client window visible and NOT let the user do anything to it. So fine - I don't hide the client. But when/if the user clicks the client window it makes that window active and I can't stop this. Nor can I make the server active again (unless the user clicks on it of course).

There is a Windows API function called EnableWindow that will work in this scenario:

```
linkdll "user"  
    int EnableWindow (word,int)  
end
```

If you want to disable/enable a window, you first need to get the other instance's window handle in your controlling instance:

```
getremote "syswindowhandle" application "toolbook" topic "deadinst.tbk"  
set vRemoteHandle to it  
get EnableWindow (vRemoteHandle, 0) --turn it off  
.  
. do your processing here  
.  
get EnableWindow (vRemoteHandle, 1) --turn it back on
```

This disables all mouse and keyboard input to the window and its children. So--don't send DDE to the window that causes it to bring up a dialog box (ask, request, tbkdlg, etc), because THEY won't get input either!

Setting Properties Of Dialog Buttons

I have created a simple dialog box using dialog.tbk; a list box, an OK button and a cancel button. Now I want to read a value from the listbox using getValue and test the value using "it", i.e. if it = ...and then go to a page based on the value of "it". When you select a field in the list box and press OK it should go to another page, but it doesn't. Does anyone have any ideas?

The problem is that the OK button was not set as TRUE/ACCEPT. The easiest way to test this is to try double clicking on a line in the single select listbox in your dialog. This will return the line that was selected no matter what the OK button is set to. If you do this and the script works, the problem is most likely the OK button.

To fix the OK button you can copy the button back to the Dialog editor and choose Edit dialog of selection from the Dialog menu. Once you have done this, double click on the OK button to change its attributes. Under the default value, choose TRUE/ACCEPT. But be careful, because when you create a new dialog button, you loose all of the script you had added to the original button.

Step by step:

1. copy the button that has the dialog script in it to the clipboard
2. start the book DIALOG.TBK
3. paste the button into this book
4. move the button somewhere off the white dialog area
5. choose "Edit dialog of selection" from the Dialog menu
6. double click on the OK button
7. change the default value to TRUE/ACCEPT
8. create a new dialog button
9. merge the script from your original button into the new button

Remember that the default behavior of push buttons in Dialog.TBK is cancel.

SIMILAR QUERY:

How do you have it take <Enter> as "ok", so that one does not have to type in the text in a field, and then use the mouse to press OK?

In the properties dialog box for push-buttons there is a box labelled "Default Value". Select "True/Accept" if you want this to be the default response (the response when the user presses <Enter>). Select "False/Cancel" if you want this to be the button selected when the user presses <Esc>.

Dialog Boxes - General Overview

Basically, you run the dialog.tbk book and use the provided utilities to design a dialog box. Be sure to "name" objects that you later might wish to set or get the value (ie: text, etc...) of. You then choose "create dialog button" from a menu - this creates a button and puts it in the CLIPBOARD. The book then tells you to go to the application you want the dialog box in and paste the button there. There is only a few lines of code in this button's script - it has a buttonUp handler. This code (or similar versions of it) can be run from anywhere. The fact that you are dealing with a button is not important. The only important thing about the button is its dlgInit property - which defines how the dialog box will appear and function. This property is created by dialog.tbk and is probably changeable (I'm not sure - haven't tried it). You can hide this button out of the way somewhere and just reference its dlgInit property or use it directly as it is a fully functional button.

To invoke the dialog box you just get the dlgInit property of the button and call the dialog function:

```
get dlgInit of <the_button>
get dialog(dlgBox of <the_button>, <dlgInit_of_the_button>)
```

This is what the default CODE in the button's script does. The dialog function returns some data that defines the STATE of the dialog box when the user leaves it. Try printing this data - its just a list. You can then use the getValue function to dissect this return value from the info you need:

```
get getValue(<return_value_of_dialog_func>, "<name_of_object>")
```

The return value of this is the information of this object, which would be its text if it is a field, or TRUE/FALSE if it is a button. You can also set the value of the objects in the dialog box (which overrides the defaults you created it with) before calling the dialog function:

```
get setValue(<return_value_of_dialog_func>, "<name_of_object>",
<new_value>)
```

The return value of this is the same as the return value of the dialog function except the default info will be updated to the new info.

Being able to set (or change) the default settings is useful: I have a generic "answer" dialog box that displays a question and asks for input. The default question and answer is null. I set these depending on the question I want to ask and when the user returns - parse the answer to check its type and correctness, possibly re-displaying the dialog box for re-entry of the data.

Keep Dialog Box Visible And Modify ListBox Contents

I want to create dialogs which "stay", e.g. in the dialog box there is a list box and a button, and when a button is pressed the dialog box doesn't disappear but the list box gets updated. I downloaded newdialog.tbk and don't understand how to use. Any help would be greatly appreciated.

You can use the NEWDLG.TBK from asymetrix.com (FTP site). It has the ability to send messages back to toolbook without closing. There is a dialog function called SETVALUE that you can use to dynamically change the values of a control.

So if you want to add a value, I'd store the list of translations in a property:

```
set my translationlist to "Fairweather--Scholastic Miscellany", & \  
                        "Colledge--14th C Texts", & \  
                        "Walsh--English Mysticism"
```

Then when you want to add a text, you can do it with:

```
push "Leclerque--Unpublished" onto my translationlist
```

Then (assuming that you have called the initialization property dlgInit) you can set up the dialog with:

```
set vInit to my dlgInit  
set vInit to setvalue(vInit,"listbox translation",my translationlist)  
.  
set vResult to Dialog(vInit, my dlgBox)
```

tbkdgl.dll., if memory serves me, the limit is about 8K in a list box.

Setting Initial Focus

How do you have a field with it ready to type in, so that you do not have to click with the mouse inside the field to begin entering text? (Highlight or focus on the field in the dialog box)

When you are creating the dialog box, you can set the layer of each of the controls (in that controls "properties" dialog box which you can bring up by double clicking the control). Set the layer of the object you want to have the initial focus to 1. <TAB> works to move between objects - set the layers of all objects capable of receiving the focus to a suitable order so that when the user tabs it makes sense.

Newdialog.tbk Features

The latest version of dialog.tbk has some features, not documented in the manual, that let you do nice things. Some examples:

1. You can link controls. For example, you can set it so that if text gets entered in a field, a button becomes active.
2. You can find out what's going on in the dialog box, while it's being displayed, from within a script..
3. You can set buttons so they DONT make the dialog box disappear.

Runtime ToolBook Errors

If I try set sysDrawDirect to false it works fine on my machine but causes an error (Runtime Toolbook can't run this command) when used with Runtime Toolbook in the lab.

Add this line to the [ToolBook] section of your win.ini file.

```
startupDrawDirect=false
```

Here is a list of commands which cause errors in Runtime Toolbook:

align	importGraphic	sort
author	pageProperties	startRecording
backgroundProperties	palettes	stopRecording
bookProperties	properties	sysGrid command
reshape	sysGridSnap	grid
rulers	sysGridSpacing	

Avoiding Repetitive Tasks

I use ToolBook to prepare presentations for class; much of this work requires repeated use of fields, all with `borderStyle=none`, `activated= true`, `fontFace=Arial`, `fontSize=16`, etc. I can put a handler in a sysbook which sets all of these, allowing me to call that routine every time I create a new field, but it doesn't seem like I should have to.

When an object is created, it is sent a MAKE message which the object can respond to--or it can allow that message to pass up its inheritance hierarchy (like all the way to a sysbook). Since there is always "one more property to set (`borderStyle`, `activated`, etc)" you use a MAKE handler in a sysbook to control these properties. To solve your problem, put this script in a sysbook that you have included in `startupSysBooks`:

```
to handle make
  system newColor
  if object of target is field
    set borderStyle of target to none
    set activated of target to true
    set fillColor of target to newColor
  end
  forward
end
```

Unchangeable But Selectable Field

Put this in the script of the field, make sure the field is activated. This script fakes the selection of text in an activated field.

```
to handle buttonDown
  system s_chrOffset
  set focus to null
  get textFromPoint(sysMousePosition)
  if item 1 of it <> -1
    set s_chrOffset to -(charCount(textline item 1 of it of text of self)
- item 2 of it)
    step i from 1 to item 1 of it
      increment s_chrOffset by charCount(textline i of text of self)
    end
    increment s_chrOffset by (item 1 of it - 1)*2
  end
end

to handle buttonStillDown
  system s_chrOffset

  get textFromPoint(sysMousePosition)
  if item 1 of it <> -1
    set newOffset to -(charCount(textline item 1 of it of text of self) -
item 2 of it)
    step i from 1 to item 1 of it
      increment newOffset by charCount(textline i of text of self)
    end
    increment newOffset by (item 1 of it - 1)*2
    select characters s_chrOffset to newOffset of text of self
  end
end

to handle mouseEnter
  set sysCursor to 3
end

to handle mouseLeave
  set sysCursor to 1
end
```


Asymetrix through its Developer Services Program offers some developers utilities and support for a yearly subscription of \$595. Some of the utilities are BookLook which allows you to look at the various type of properties in your book. Searcher which allows you to search for just about anything in your book. Remover which allows you to visually remove your script from your book, the process is transparent because your book works as if the script is there however if a hacker should try to edit your book he/she won't be able to see the script. There is also a simple install utility for automating the installation of a Toolbook application that you distribute.

Manipulating Scripts In Runtime ToolBook

Is there a way of assigning new scripts to objects within runtime version? What I am trying to do is to give the user the opportunity to create his own hotwords of his individual texts without accessing to author mode.

You can't set scripts in runtime mode, but you can create hotwords.

What you could do, is write a generic script for hotwords and put it in the book script (or wherever is appropriate for your application).

Example:

Lets say that you want to navigate to a page whenever the hotword is clicked on.

Here's the script to set up the hotword:

```
to handle make
  if object of target = "hotword"
    ask "What page is associated with this hotword?"
    if it is not NULL
      set navPage of target to it
    else
      send removeHotword
    end
  else
    forward
  end
end
end
```

Here's the script to navigate with:

```
to handle buttonUp
  if object of target = "hotword"
    if navPage of target is not NULL
      go page (navPage of target)
    end
  else
    forward
  end
end
end
```